Integrating low-level motion cues in deep video saliency

Pol Caselles Rico

Department of Signal Theory and Communications,
Engineering of Technologies and Services of Telecommunication
(ETSETB)

Polytechnic University of Catalonia

Supervisor

Kevin McGuinness and Xavier Giró-i-Nieto

In partial fulfillment of the requirements for the degree of

Bachelor of Science in Engineering in Electrical Engineering

June 24, 2019

Acknowledgements

I would like to thank my supervisors, Xavi Giró-i-Nieto and Kevin McGuinness, for all the guidance and help provided during the realization of the thesis. Also, I would like to thank Enric Moreus, and Juanjos help because while I was learning deep learning they helped me having a well-organized project using the newest tools.

Abstract

This thesis investigates the importance of motion when predicting saliency in videos. Naturally, humans observe both dynamic and static objects. When we are focused on watching a video, we tend to keep our eyes on the objects that are moving in the scene, items that we quickly recognize, as well as to those that attract our attention. In this work, different experiments are presented to corroborate this implication. Various approaches will be shown implementing an adaptation of the SalBCE neural network by using only motion. A simple implementation is proposed for the generation of saliency maps using previously extracted static and dynamic information from the images. The DHF1K dataset has been used for the experiment's realization.

Abstract

Esta tesis investiga la importancia del movimiento al predecir el saliency en los videos. Naturalmente, los humanos observan objetos tanto dinmicos como estáticos. Cuando nos enfocamos en mirar un video, tendemos a mantener la vista en los objetos que se mueven en la escena, los elementos que reconocemos rápidamente, así como aquellos que atraen nuestra atención. En este trabajo, se presentan diferentes experimentos para corroborar esta implicación. Se mostrarán varios enfoques implementando una adaptación de la red neuronal SalBCE utilizando solo movimiento. Se propone una implementación simple para la generación de mapas de saliency utilizando información estática y dinámica extraída previamente de las imágenes. El conjunto de datos DHF1K ha sido utilizado para la realización del experimento.

Abstract

Aquesta tesi investiga la importància del moviment en predir el saliency en els vídeos. Naturalment, els humans observen objectes tant dinàmics com estàtics. Quan ens enfoquem en mirar un vídeo, tendim a mantenir la vista en els objectes que es mouen en l'escena, els elements que reconeixem ràpidament, així com aquells que atreuen la nostra atenció. En aquest treball, es presenten diferents experiments per corroborar aquesta implicació. Es mostraran diversos enfocaments implementant una adaptació de la xarxa neuronal SalBCE utilitzant sol moviment. Es proposa una implementació simple per a la generació de mapes de saliency utilitzant informació estàtica i dinàmica extreta prèviament de les imatges. El conjunt de dades DHF1K ha estat utilitzat per a la realització de l'experiment.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Project overview and goals

Saliency prediction is a topic studied in computer vision field. It consists of predicting the zones of an image/video that people will give more importance. Visual attention enables humans to quickly analyze complex scenes by giving to higher regions of processing and visual awareness. This behavior can be modeled as Saliency Prediction and we can also apply this concept to videos. In order to represent the saliency, we usually generate heat maps where each pixel has a value from 0 to 1, meaning less or more odds to be attracted by the human eye. We have to take into account that making these saliency predictions in videos we increase the difficulty from static images. The main hypothesis is that people tend to give more importance to objects that are moving, so, for this reason, a salient heat map of an image could differ from the same image as a frame of a video.

In order to calculate and predict the saliency, we have chosen to use DNN (deep neural networks[1]) and machine learning as a basis to face the challenge. We have to take into account that, in the official benchmarks (DHF1K[2] dataset), the ones that give the best score are those based on these technologies. For this reason, we have decided to implement certain improvements in the architecture created by my partner Juanjo, SalBCE[3]. The main idea is to add the motion within the saliency prediction process, due to the strong hypothesis that the movement has huge importance. To carry out this architecture, the SalBCE model has been used to generate salience maps only from the RGB images. On

the other hand, the same model has been made inference on the images associated with the motion of these. From the two groups of images, an own model has been used, in order to integrate the static and dynamic information to generate the final saliency maps.

The main objective of this work is to discover the importance of movement in the prediction of saliency, and if that importance is proven, find a solution integrating this concept. In order to carry out this process, the following objectives have been proposed during the execution of the work:

- Check the importance of movement in saliency;
- Extract the information related to the movement efficiently;
- Propose a solution that integrates the previous information;
- Obtain better results than the non-movement approach;

1.2 Work Plan

For the completion of the end-of-grade project, weekly meetings have been planned with my two tutors (Xavier Giró-i-Nieto and Kevin McGuinness) in order to discuss all the steps taken and plan the work to be done. We have based on a system of micro-tasks where each week the work to be done is defined for the following week.

The realization of this work has been carried out in the Insight Center for Data Analytics[1] research center located at Dublin City University (DCU). The meetings have been done through video calls with the two tutors in almost all the meetings. Because each university has a different schedule for the presentation of the documentation and delivery of the work, the following schedule is defined:

1. Arrival at DCU (Dublin), 23.01.2019
2. Initiation meeting. Objectives presented: 27.02.2019
3. Status Report: 28.02.2019
4. Presentation Submission: 12.03.2019
5. Report Submission: 21.05.2019
6. Poster Submission: 21.05.2019
7. EXPO Attendance: 23.05.2019
8. Continuation of the project at the UPC (Barcelona): 25.05.2019
9. Completion of the project: 25.06.2019

[1] https://www.insight-centre.org/

1.3 Work Breakdown Structure

Below is a diagram summarizing the general line of work carried out in this project in order to facilitate the understanding of the different tasks carried out.

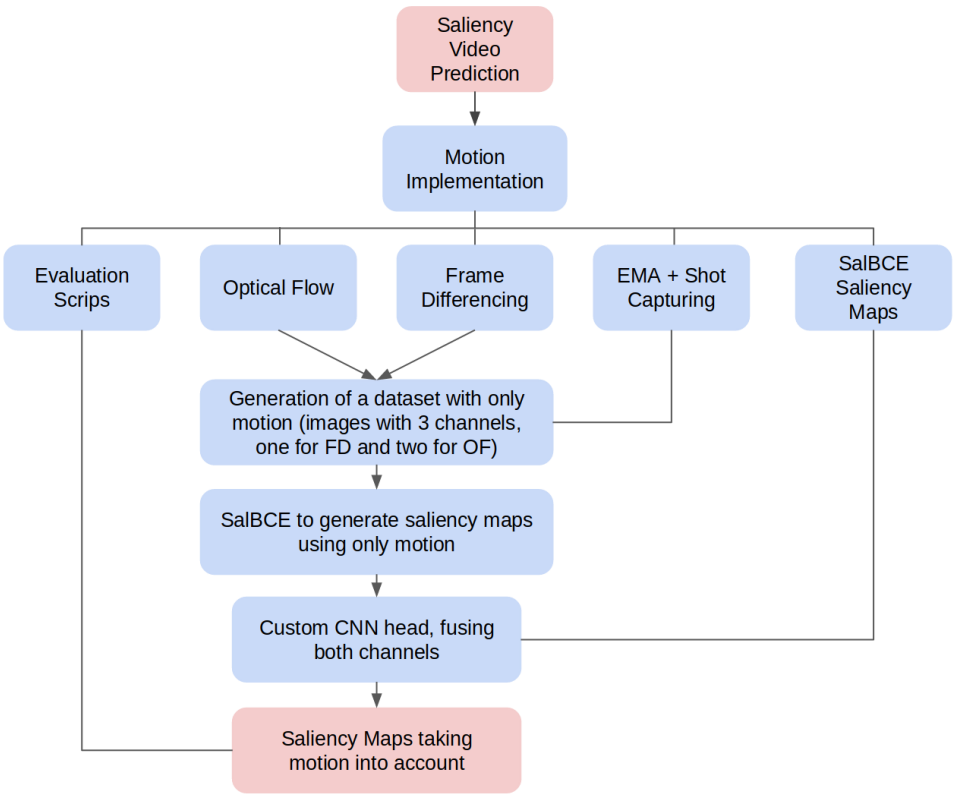


Figure 1.1: Work Breakdown Structure of the project

In Figure 1.1 the general structure of the work is shown. Before start developing the main line of work, it was necessary to carry out evaluation scripts in order to be able to apply the metrics, and thus be able to compare the results. On the other hand, it was necessary to use SalBCE to generate the saliency maps without the motion.

During the progression of the work different scripts have been tested as well as the verification of the results in order to verify the initial hypothesis. At first, the motion of the dataset under study (DHF1K) has been computed, and we store new images with this information. Next, EMA has been applied to these new images in order to store information from the past. Using Custom CNN head , static maps have been combined with maps that take into account the motion to generate a final saliency map.

1.4 Milestone list

Tasks	Description	Date
State of the art	Realization of the module in Machine Learning to begin to obtain concepts about Computer Vision. Read and study the work that Juanjo did. Study the appropriate papers to understand the basis of the project.	14.02 - 28.02
Become familiar with the environment	Work with Juanjo to understand his code and how to execute it in the work environment of the Insight Center. Become familiar with Machine Learning concepts.	27.02 - 01.03
Code, test, verify the Optical Flow	Read the documentation to learn the Optical Flow concept. Compute the Optical Flow of the entire dataset.	04.03 - 11.03
Compute frame differencing	Implement the difference of frames as another method to measure the movement. Apply the EMA taking into account the change of scene.	4.03 - 8.03
Generation of datasets	Generate images where the three channels corresponding to the RGB, the magnitude of the difference of frames are stored, in the other two channels the position x and y of the Optical Flow.	11.03 - 28.04
Comparison	Train the Juanjos model with the images as inputs where only the movement between frames is taken into account. Frame differencing and Optical Flow	08.04 - 01.05
A fusion of the two models	Implement a simple structure that has as inputs the saliency maps generated with the static information of the images and the Optical Flow obtained with SalBCE.	01.05 - 20.05
One stream approach	Implement SalBCE that has as inputs the saliency maps generated with the static information of the images and the dynamic information	30.05 - 15.06

Table 1.1: Milestone list

1.5 Time Plan (Gantt diagram)

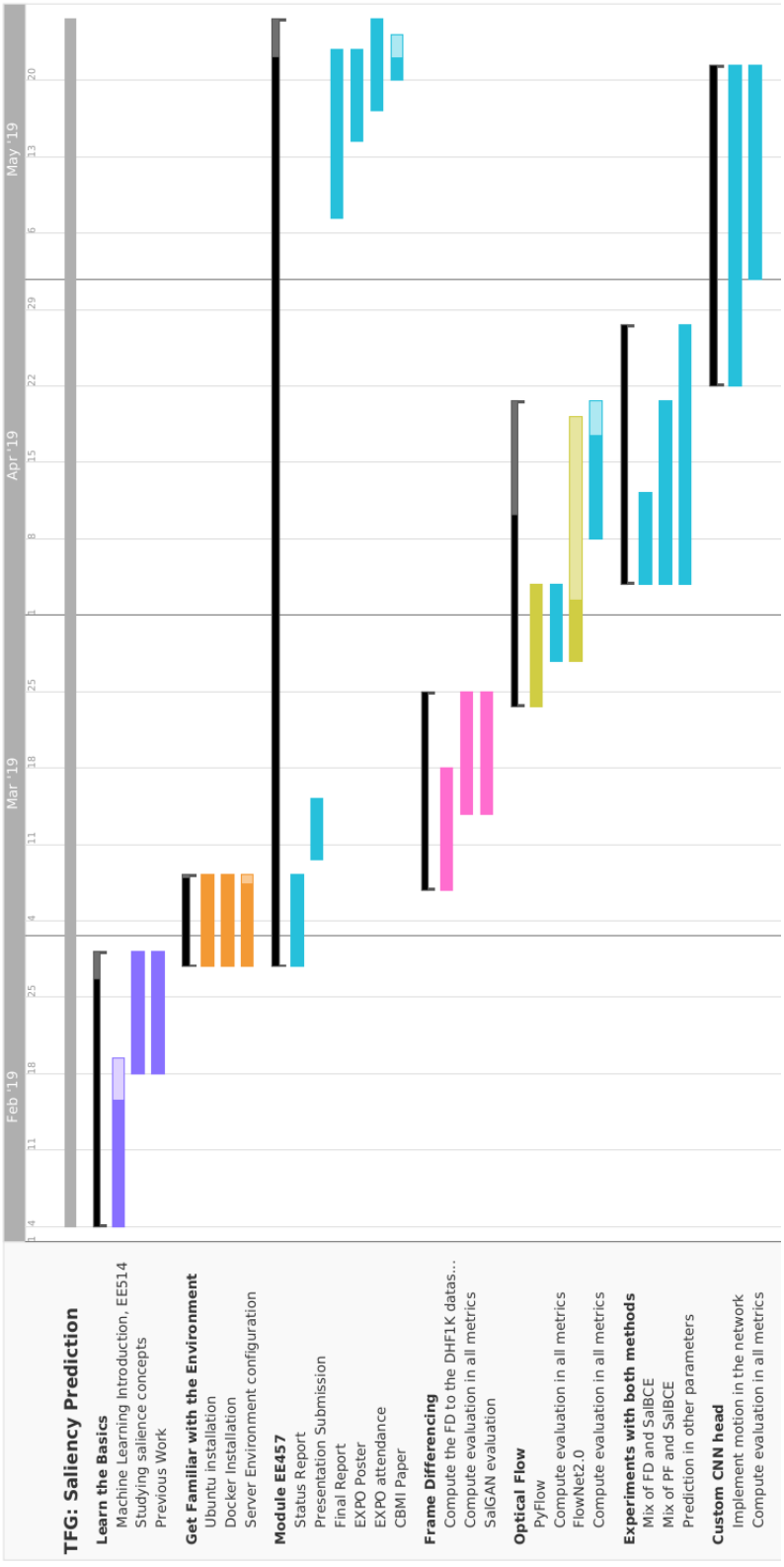


Figure 1.2: Gantt Diagram of the project

Chapter 2

State of the art

Summary

The background concepts will be described as well as the necessary information to understand why the initial hypothesis and how to start developing the idea. The architectures used will also be presented, the dataset from which the whole study has been developed as well as the breakdown of the metrics used with their respective mathematical development. The aim is to introduce the initial concepts that led to the beginning of the basic idea of the importance of motion in the prediction of saliency.

2.1 Technical Background, Architecture used

2.1.1 SalGAN

Visual Saliency Prediction with Generative Adversarial Networks (SalGAN[4]) model was proposed as an adversarial training for visual salience prediction (GAN) instead of on-adversarial training (BCE) and even combine them to obtain better results in stability and convergence rate. In this study, they proposed the SalGAN model for visual saliency prediction. They concluded that the proposed GAN training approach is generic and that using its structure they could apply to improve the performance of other saliency models.

SalGAN is composed of two parts; the generator that has a convolutional encoder-decoder (the encoder consists of a fully convolutional architecture based on the popular VGG-16[5] and the decoder with the inverse layers of the encoder, replacing max-pooling with up-sampling layers) and the discriminator that is in charge of distinguishing between ground truth saliency maps and generated saliency maps from the generator.

During adversarial training, the loss function of the saliency is a combination between the error from the discriminator and the cross-entropy with respect to the ground truth. As they show, a $\alpha = 0.1$ has good results.

$$\mathscr{L}_{BCE} = -\frac{1}{N}\sum_{j=1}^{N}(S_j log(\hat{S}_j) + (1 - S_j) log(1 - \hat{S}_j))^2 \tag{2.1}$$

$$\mathscr{L} = \alpha \mathscr{L}_{BCE} + L(D(I, \hat{S}_j), 1) \tag{2.2}$$

During the training of the discriminator, no content loss is available and the loss function is:

$$\mathscr{L}_D = L(D(I, S_j), 1) + L(D(I, \hat{S}_j), 0) \tag{2.3}$$

2.1.2 SalBCE

SalBCE[3] is an adaptation of SalGAN architectures with some changes. It consists of using only the generator part, using the Binary Cross Entropy (BCE) as the loss function. In addition to training using RGB, it is included depth as well as CoordConv. In the first case, it was based on the idea that the objects closer are more salient than those that are further away. In the second case, the idea of this technique is to provide the convolutional input layers access to its own input coordinates through the use of extra coordinate channels. Finally, the idea of implementing the Optical flow was proposed, although it was not developed in his work.

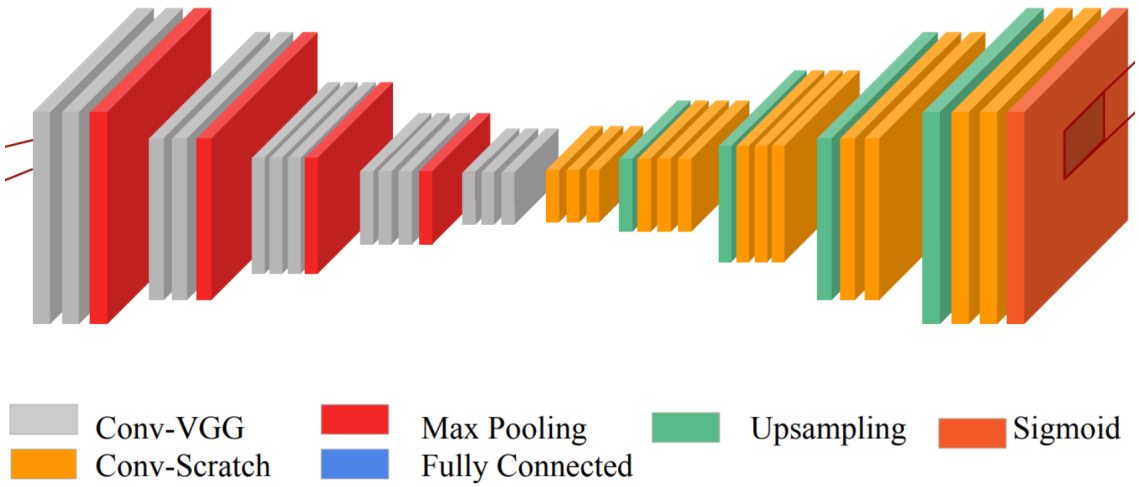


Figure 2.1: SalBCE model architecture

In figure 2.1 it can be seen the architecture used, with a total of 31,787,009 hyperparameters: During the training part it has been using the Adam [6] optimizer. The initial weights have been obtained from SalGAN. Starting with a learning rate of 0.00001 and dividing by 10 every 3 epochs (total of 27 epochs) with a batch size of 12 (due to the limitation of space in the GPU) has been proposed. For the baseline weights, SalBCE has trained on SALICON[7] dataset, which the images used are from the COCO[8] dataset. These weights are going to be the ones used as a baseline in our custom CNN head architecture based on SalBCE.

2.1.3 ACLNet

ACLNet is a Convolutional Neural Network (CNN), model designed to predict saliency from videos. More recently, they presented a new video saliency prediction benchmark. They propose a novel deep learning based video saliency model (Figure 2.2), which encodes to supervised attention mechanism to explicitly capture static saliency information and help LSTM better capture dynamic saliency representations over successive frames:

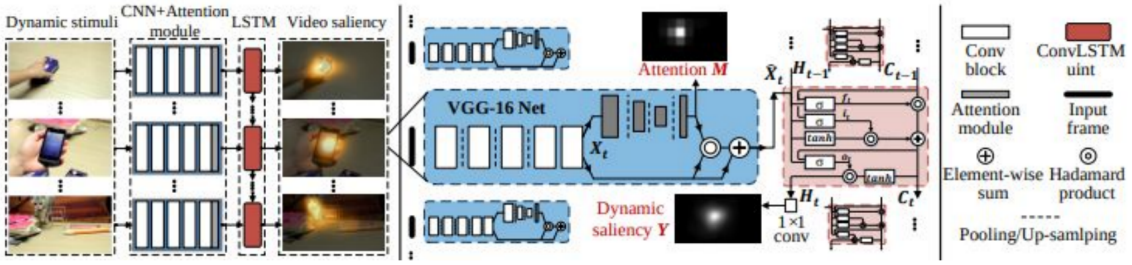


Figure 2.2: ACLNet architecture composed of VGG-16, Attention module and ConvLSTM.

In their model, they used an attention mechanism in order to obtain the static information to leave the LSTM focus on the temporary information. The attention module was trained in the SALICON dataset, while the whole architecture was trained in a combination of different datasets, DHF1K[2], HollyWood2[9] and UCF Sports[10]. They used as a loss function a combination of three different metrics, in order to be able to measure more accurately depending on the different proposed metrics (in SalBCE, BCE was used as a loss function).

$$\mathscr{L}(Y,P,Q) = \mathscr{L}_{KL}(Y,Q) + \alpha_1\mathscr{L}_{CC}(Y,Q) + \alpha_2\mathscr{L}_{NSS}(Y,P)) \tag{2.4}$$

where the predicted saliency map is $Y \in \{0,1\}^{28\times28}$, the map of fixation locations is $P \in \{0,1\}^{28\times28}$ and the continous saliency map (distribution) is $Q \in \{0,1\}^{28\times28}$. Note that in order to obtain the Q continous saliency map, a blurring with a small Gaussian Kernel must be applied to the discrete fixation map P. αs are balance parameters ans are empirically set to $\alpha_1 = \alpha_2 = 0.1$

2.1.4 SalBCE motion approach

Our Custom CNN head is based on the motion integration together with the result of SalBCE of the RGB images. To improve the prediction using only the dynamic information, at first, optical flow (OF, see in 3.2) and frame differencing (FD, see in 3.1) are computed, and using a structure of one stream architecture [11] an image is generated, in which the three channels correspond to $[FD,\ OF_{x\ axis},\ OF_{y\ axis}]$. Then, using the SalBCE architecture, the saliency maps are generated only from the images that contain motion's information.

Parallel to this, the saliency maps are also generated from the RGB images, using the SalBCE architecture. From this point, an image of 2 channels corresponding to the previous output is generated. Where the first channel corresponds to the saliency map of the static images and the other channel to the dynamic information of those. A new architecture of 4 convulsion layers (Table 6.1) is presented, which finally generates a saliency map integrating all the information.

Layers	depth	Kernel	Stride	Pad	Activation
conv2d	64	3x3	1	1	ReLU
conv2d	256	3x3	1	1	ReLU
conv2d	128	3x3	1	1	ReLU
conv2d	64	3x3	1	1	ReLU
Output	1	1x1	1	0	Sigmoid

Table 2.1: Fusion saliency maps architecture

SalBCE's baseline weights have been used (SalBCE trained on SALICON for 27 epochs), and SalBCE has been trained in RGB images for 4 epochs, and 20 epochs for the dynamic images. Making inference in the images which only have motion information, it is not enough to overcome SalBCE on RGB metrics. However, the results obtained with the PF and OF images are close to it, only in the AUC-shuffle, it obtains better results.

In order to train the entire architecture, we have trained every dataset separately (GPU limitations). SalBCE's baseline weights have been used (SalBCE trained on SALICON

for 27 epochs), and SalBCE has been trained in RGB images for 4 epochs, and 20 epochs for the dynamic images. The learning rate started at $1 \cdot 10^{-5}$ and every 2 epochs we divide it by 10, and we used an Adam optimizer. Once we have generated both saliency maps for the entire train DHF1K dataset, we fuse both maps for each image $I^j_{mix} = I^j_{From\ RGB} + I^j_{from\ motion}$ as an input of the architecture in Table 6.1. (The results of this implementation are in 4.4)

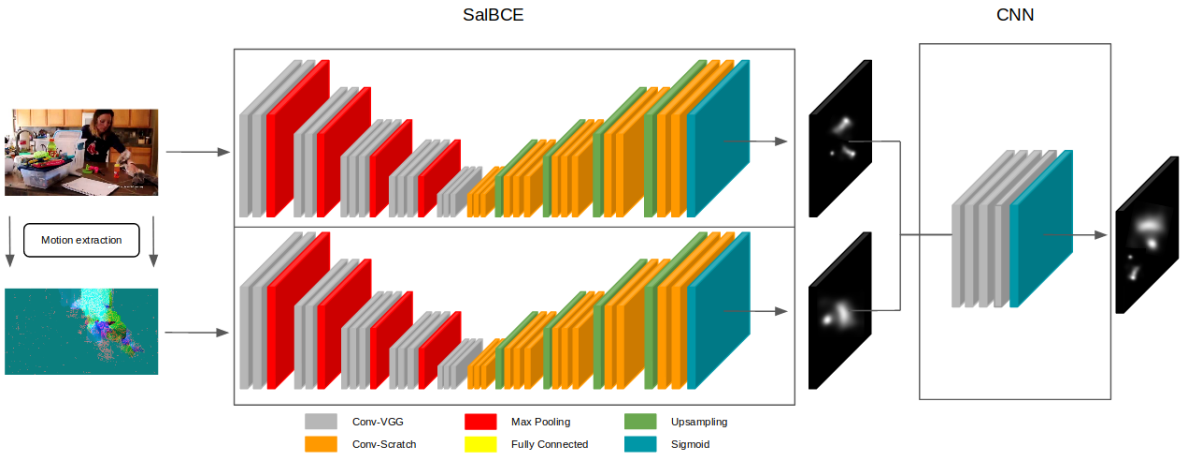


Figure 2.3: The full architecture used. Two SalBCE in parallel, and a custom CNN head.

2.1.4.1 One stream approach

In order to simplify the training process and simplify the model, the use of SalBCE is only proposed by inputting 6 channel image as input (the RGB image and the motion information) $[R, G, B, FD,\ OF_{x\ axis},\ OF_{y\ axis}]$.

Training with the same parameters as in the 2.1.4 section, the Custom Neural Network is thus eliminated and the training time of the model is halved. Important train from scrath because the model was pre-trained only with the images of 3 channels. If the weights are reused the system will tend to eliminate the information related to the movement.

2.2 Working Dataset

In this thesis, we have exclusively used the dataset called DHF1K. This dataset stand from Dynamic Human Fixations 1K it is the largest video saliency dataset. As we can see in Table 2.2, this dataset is the newest one. It contains 1000 videos, which the first 600 are dedicated to training, from 601 to 701 for validation and the last 300 for the test. For the generation of this dataset, they searched in the Youtube platform for around 200 keywords and selected 1000 video sequences. The videos were exported at 30fps at a resolution of 640x360.

Dataset	Year	Videos	Resolution	Duration(s)	Viewers	Task
CRCNS	2004	50	640x480	6-94	15	task-goal
Hollywood-2	2012	1,707	720x480	2.120	19	task-goal
UCF sports	2012	150	720x480	2-14	19	task-goal
DIEM	2011	84	1280x720	27-217	∼50	free-view
SFU	2012	12	352x288	3-10	15	free-view
DHF1K	2017	1,000	640x360	17-42	17	free-view

Table 2.2: Statistics of typical dynamic eye-tracking datasets

This set of videos stores a total of 150 different categories. It also offers diversity with different camera movements and content. To generate the Ground Truth of these videos they used an eye-tracker called Sensoc Motoric on a total of 17 participants of an age between 20 and 28 years old. Note that the image size is higher than the input of the above architectures, for this reason, a resize will be applied to use them.

2.3 Metrics used

In order to be able to measure how well or badly predictions are made, we need part of the ground truth to compare the correct solutions. The mathematical function that we apply between the reference and the predicted map is called system metrics and they allow us to evaluate the distance with respect to the solution. Although there is a large number of metrics available, only the 5 most popular ones have been used. As it is shown in this paper [12], the result could be optimized for each of these metrics, however, only one prediction has been generated for each frame. Following is a brief summary of the metrics used [13]:

Location-based metrics

Given the goal of predicting the fixation locations on an image, a saliency map can be interpreted as a classifier of which pixels are fixated or not. This suggests a detection metric for measuring saliency map performance.

- AUC_Judd: This metric consists of calculating the area under the ROC curve. To draw the ROC curve, only true positive rate (TPR) and false positive rate (FPR) are necessary. The TPR defines how many positive positives results occur among all positive samples available during the test. FPR, on the other hand, defines how many incorrect positives results occur among all negative samples available during the test.

$$TPR = \frac{TP}{P} = \frac{TP}{TP + FN} \qquad FPR = \frac{FP}{N} = \frac{FP}{FP + TN} \tag{2.5}$$

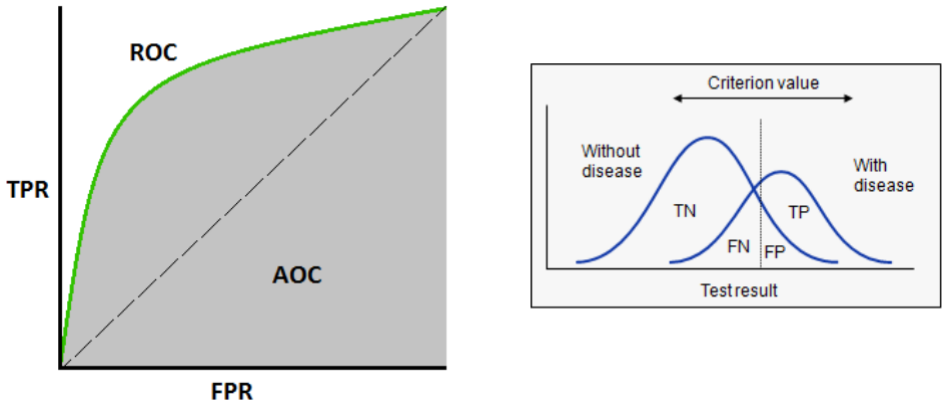


Figure 2.4: ROC curve and its density function representation

The maximum value of this metric (coincides with the best possible value) is 1, coinciding that to get a 100% of true positives there are a 0% false positives (Ideal case where the two probability functions are sufficiently separated so as not to overlap at any point). As we can see in the image above, the ROC curve identifies the relationship between true positives and false negatives in the entire possible spectrum of the threshold value.

AUC is computed by varying the threshold of the saliency map and computing to a trade-off between true and false positives. Lower thresholds correspond to measuring the coverage similarity between distributions, while higher thresholds correspond to measuring the similarity between the peaks of the two maps.

- AUC-Shuffled: In general, datasets tend to include a higher density near the center of the image. Taking into account this, making a model where only the fixations appear in the center will obtain greater results than if we did it in a random way. The shuffled AUC metric, sAUC samples negatives from fixation locations from other images, instead of uniformly at random. This has the effect of sampling negatives predominantly from the image center because averaging fixations over many images results in the natural emergence of a central Gaussian distribution.

- NSS: The Normalized Scanpath Saliency consists of measuring the average of the saliency map predicted in those points wherein the ground truth there is a fixation point. For this reason, the value for each pixel of the fixation ground truth map $Q^j \in \{0, 1\}$ is binary, making the average only in certain regions of the image. Before computing this metric, the predicted P map is normalized with respect to the mean, so if there is a false positive in an area where there is no fixation point, it is reducing the value of the others that are in the correct area, for this reason, NSS is sensitive to false positives.

$$NSS(P, Q^B) = \frac{1}{N} \sum_j \overline{P_i} \cdot Q_j^B \tag{2.6}$$

$$N = \sum_j Q_j^B \quad , \quad \overline{P} = \frac{P - \mu(P)}{\sigma(P)} \tag{2.7}$$

  The mean saliency value is subtracted during computation, NSS is invariant to linear transformations like contrast offsets. Where i indexes the i^{th} pixel, and N is the total number of fixated pixels. Positive NSS indicates correspondence between maps above chance, and negative NSS indicates anti-correspondence.

Distribution-based metrics

The (location-based) metrics described so far score saliency models at how accurately they predict discrete fixation locations. If the ground truth fixation locations are interpreted as a possible sample from some underlying probability distribution, then another approach is to predict the underlying distribution instead of the fixation locations

- SIM: The similarity metric can be understood as the computation of the *min* function between the saliency prediction map P and the fixation map Q^D for each pixel, and then the average of these. The saliency prediction map and the binary eye fixation map are normalized:

$$SIM(P, Q^D) = \sum_j min(P_j, Q_j^D) \tag{2.8}$$

$$where \quad \sum_j P_j = \sum_j Q_j^D = 1 \tag{2.9}$$

  iterating over discrete pixel locations i. A SIM of one indicates the distributions are the same, while a SIM of zero indicates no overlap. SIM is very sensitive to missing values, and penalizes predictions that fail to account for all of the ground truth density.

- CC: Pearson's Correlation Coefficient can be understood as an indicator that expresses how two variables are linearly correlated, or in another way, how dependent is one of the other. CC can be used to interpret saliency and fixation maps, P and Q^D:

$$CC(P, Q^D) = \frac{\sigma(P, Q^D)}{\sigma(P) \cdot \sigma(Q^D)} \tag{2.10}$$

  where (P, Q^D) is the covariance of P and Q^D. CC is symmetric and penalizes false positives and negatives equally. High positive CC values occur at locations where both the saliency map and ground truth fixation map have values of similar magnitudes.

To obtain good results in the previous metrics, it is necessary to take into account how the metrics vary according to the saliency maps generated. As shown in Table 2.5, it can be seen which properties affect and which do not in each type of metric. For example, in

the variation of the variance in the prediction has no implications in the AUC metric but a huge descend appear in the SIM metric.

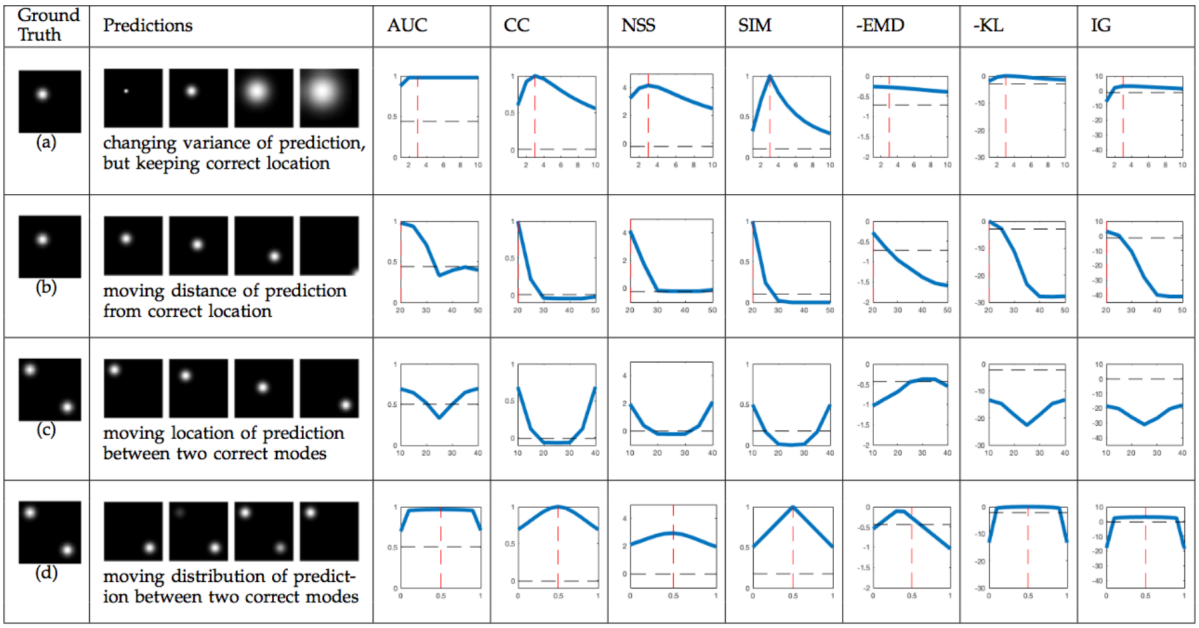


Figure 2.5: Variation of metrics selected for saliency prediction evaluation

In this study [12] they propose the generation of different saliency maps to improve as much as possible in each of the metrics. In this work, we are not going to continue in this direction and we will propose a single map for each prediction. Below are three ways to normalize the saliency maps, and depending on the type required will apply one or the other as appropriate in each circumstance.

$$Norm_{Range} \quad S \longrightarrow \frac{S - min(S)}{max(S) - min(S)} \tag{2.11}$$

$$Norm_{Variance} \quad S \longrightarrow \frac{S - \mu(S)}{\sigma(S)} \tag{2.12}$$

$$Norm_{Sum} \quad S \longrightarrow \frac{S}{sum(S)} \tag{2.13}$$

Chapter 3

Techniques studied

Summary

Below are the different concepts used for the realization of the thesis. The mathematical bases are defined to perform the tasks of Frame Differencing and Optical Flow as well as the development used to implement the exponential Moving average and the shot detection. A brief summary of certain basic concepts of statistics is also presented. It is intended to give sufficient development to understand in detail all the processes carried out subsequently correctly.

3.1 Frame Differencing (FD)

Frame Differencing is a technique used in video whose objective is to obtain the motion of the image by calculating the difference between one frame and the next. This technique consists of measuring the Euclidean distance between the vector with RGB components of each pixel with respect to that same pixel of the next frame. In this way, for every two images we will obtain one of the same resolution, in which each pixel will have a unique value, resulting from the following formula:

$$O_j^t = \left\| x_j^t - x_j^{t+1} \right\|_2 = \sqrt{(r_1 - r_2)^2 + (g_1 - g_2)^2 + (b_1 - b_2)^2} \tag{3.1}$$

O_j^t : Heatmap resulting from the frame differencing at frame t, one channel per pixel

$x_j^t = [r, g, b]^T$: Full image with 3 channels per pixel

$r_{1,2}, g_{1,2}, b_{1,2}$: red, green, blue channels of each pixel

This method has a great capacity to detect moving objects, as long as the scene remains static (that is, the camera must remain still, without displacement). It also will not work correctly if there is a scene change between two consecutive frames - it will detect a large amount of movement in the whole image due to the large difference - or even a variation of the illumination. On the other hand, the rate of frames per second can cause variations in the magnitude of the movement.

Assume the same object moving at a constant speed in a section of the video. When dividing it between 10 frames, it will have a much greater apparent speed than if we divide that same fragment of video between 100 frames. For this same reason, if the object moves at high speed and the fps rate is low, its movement cannot be detected correctly.

In order to reduce the noise, a blurring has been applied just after the calculation, the average has been computed, and all those values that are below have been forced to be worth the minimum. Then, all those pixels that had a magnitude smaller than a threshold have been eliminated and, subsequently, a stronger blurring has been applied again than the previous one. In this work the value of this threshold has not been optimized,

since one has been chosen that captures enough information. The heatmap has been normalized (See 2.11) between 0 and 1, where 0 means the absence of movement and 1 the maximum possible. It can be understood that, although two videos have objects moving at a different speed, for the purposes of this normalization (applied with respect to the maximum of each frame), there will be no difference.

Therefore, once the differencing frame is applied, we will obtain a heatmap whose values will be comprised between 0 and 1, being able to be interpreted as the zones with the most movement of each frame.

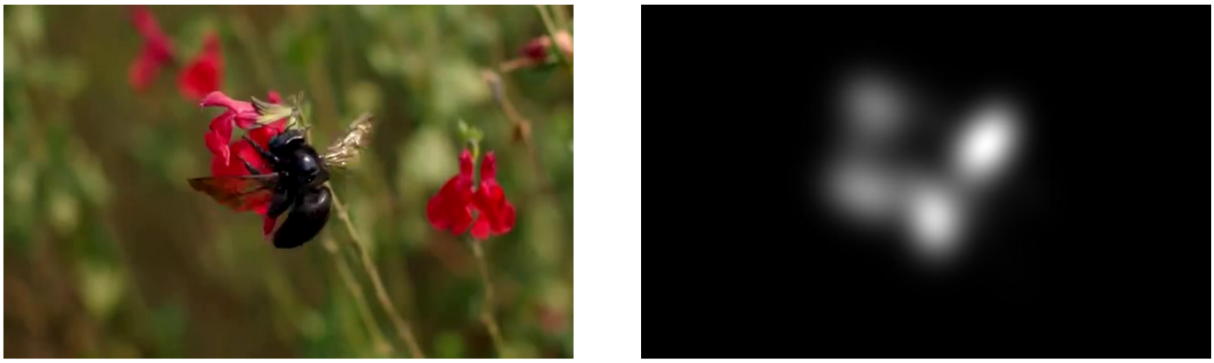


Figure 3.1: Left image is the video 007 of the DHF1K dataset, frame 99. Right image is the FD saliency map between frames 99 and 100

3.2 Optical Flow (OF)

Optical Flow can be understood as the motion vectors of each pixel in a given two images. That is, given a couple of images, to determine where each of the pixels moves between the two frames. Depending on the degree of sophistication, 2D or 3D maps can be generated. Optical Flow can also be defined as the distribution of apparent velocities of movement of brightness pattern in an image.

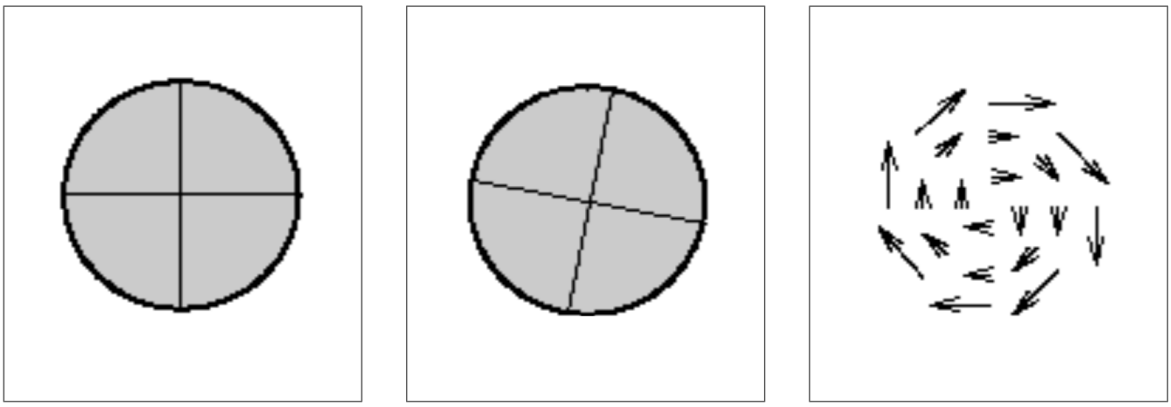


Figure 3.2: Optical Flow interpretation. Left image is the frame at time t. Middle image is the frame at time $t + 1$. Right image is the Optical Flow output.

The result of the optical flow is an image of two channels where for each pixel it expresses the direction x and y associated with the movement vector of this, between the two consecutive frames. Although it may seem a relatively easy task, there are different aspects that should be highlighted. You can define the movement in different ways: there are different factors that can influence the motion, such as the movement of the camera, the scene or the change of luminosity. In order to gain insight into methodology used to calculate the Optical Flow, below is a summary:

Lukas Kanade methodology [14]

First we define the equation of motion, where u and v are the variation for each axis between two frames:

$$I(x, y, t) = I(x + u, y + v, t + 1) \tag{3.2}$$

Take Taylor expansion of $I(x + u, y + v, t + 1)$ at (x, y, t) to linearize the right side:

$$I(x + u, y + v, t + 1) \approx I(x, y, t) + \frac{\partial I}{\partial x}u + \frac{\partial I}{\partial y}v + \frac{\partial I}{\partial t} \tag{3.3}$$

$$I_t + \nabla I[u, v]^T = 0 \tag{3.4}$$

In (3.4), there is only one equation but we need to determine u and v. This is called the aperture problem [15]. In order to understand it, we can imagine the typical barber spiral which is spinning, however, we see as it is moving upward. Therefore we impose additional constraints. We assume that the flow field is smooth locally, one method is to pretend the pixel's neighbors have the same (u, v). If we use a 5x5 window, that gives us 25 equations per pixel:

$$0 = I_t(P_i) + \nabla(P_i) \cdot [u, v]^T \tag{3.5}$$

$$\begin{bmatrix} I_x(p_1) & I_y(p_1) \\ I_x(p_2) & I_y(p_2) \\ \vdots & \vdots \\ I_x(p_25) & I_y(p_25) \end{bmatrix} \begin{bmatrix} u \\ v \end{bmatrix} = - \begin{bmatrix} I_t(p_1) \\ I_t(p_2) \\ \vdots \\ I_t(p_25) \end{bmatrix} \tag{3.6}$$

$$Ad = b \quad \longrightarrow \quad minimize \quad ||Ad - b||^2 \tag{3.7}$$

We have more equations than unknowns: solve least squares problem. This is given by:

$$A^T A \cdot d = A^T b \tag{3.8}$$

Optimal (u, v) satisfies Lucas-Kanade equation:

$$\begin{bmatrix} \sum I_x I_x & \sum I_x I_y \\ \sum I_x I_y & \sum I_y I_y \end{bmatrix} \begin{bmatrix} u \\ v \end{bmatrix} = - \begin{bmatrix} \sum I_x I_t \\ \sum I_y I_t \end{bmatrix} \tag{3.9}$$

This equation is solvable when $A^T A$ is convertible, $A^T A$ is not too small due to noise, eigenvalues λ_1 and λ_2 of $A^T A$ should not be too small, and finally $A^T A$ should be well-conditioned (λ_1/λ_2) should not be too large. The $A^T A$ matrix has the same shape as the formula for Finding Corners [16]:

$$C = \begin{bmatrix} \sum I_x^2 & \sum I_x I_y \\ \sum I_x I_y & \sum I_y^2 \end{bmatrix} \tag{3.10}$$

$$C = R^{-1} \begin{bmatrix} \lambda_1 & 0 \\ 0 & \lambda_2 \end{bmatrix} R \tag{3.11}$$

where R is a rotation matrix, so corners are the things we can track. Corners are when $\lambda 1$, $\lambda 2$ are big, this is also when Lucas-Kanade works. Corners are regions with two different directions of the gradient (at least). Aperture problem disappears at corners. To work correctly we have to assume that the movement will be small, there will be a low level of noise in the image and the chosen window will not be too large and the neighboring pixels will move in the same direction. If this is not the case, the pyramid method is proposed, where the image is resized until it has a smaller motion and then it is applied layer by layer. Even so, it will not be detected correctly with color and shape changes.

In the beginning, this implementation was started, although in the end it was opted for a coarse-to-fine optical flow method[17] which, in principle, has better results predicting the motion of the objects. The mathematical explanation of this model is summarized below:

Taking into account the same definition of grayscale sequence, where $I(x, y, t)$ is the grey value of the point (x, y) at the first frame while the gray value of the corresponding point at the next frame is $I(x + u, y + v, t + 1)$. The grey value constancy assumption is very sensitive to the illumination changes, this is the reason for the proposal of the gradient constancy assumption:

$$\nabla I(x, y, t) = \nabla I(x + u, y + v, t + 1) \tag{3.12}$$

They combined the grey value constancy assumption and the structure tensor constancy assumption to create the energy function to be minimized $x = (x, y, t)^T$ and $w = (u, v, 1)^T$:

$$E_{data}(u, v) = \iint_{\Omega} \Psi(|I(x + w) - I(x)|)^2 + \gamma\Psi(|I(x + w) - \nabla I(x)|)^2 \quad dxdy \tag{3.13}$$

$$\Psi(s^2) = \sqrt{s^2 + \epsilon^2} \quad , \qquad \epsilon = 0.001 \tag{3.14}$$

where $\Psi(s2)$ is the penalty function which reduce the influence of the outliers. To design the smoothing term, they first introduce the classical smoothing strategy proposed by Horn and Schunck [18] as follows:

$$E_{smooth}(u, v) = \iint_{\Omega} \Psi(|\nabla_3 u|^2 + |\nabla_3 v|^2) \quad dxdy \tag{3.15}$$

In the traditional optical flow method, the weighting factor of the smoothing term usually was a fixed value which made the same smoothing extent in any area of the image. With the design of the data term and smoothing term, the proposed optical flow model could be written as the followed energy function:

$$E(u, v) = E_{data} + \alpha E_{smooth} \quad \longrightarrow \quad minimize \tag{3.16}$$

The Euler-Lagrange equations [19] are nonlinear in their argument $w = (u, v, 1)^2$. The first step towards a linear system of equations, which can be solved with common numerical methods, is the use of fixed point iterations on w. In order to implement a multiscale approach, necessary to better approximate the global optimum of the energy, these fixed

point iterations are combined with a downsampling strategy. Instead of the standard downsampling factor of 0.5 on each level, it is proposed here to use an arbitrary factor $\eta \in (0, 1)$, which allows smoother transitions from one scale to the next 1. Moreover, the full pyramid of images is used, starting with the smallest possible image at the coarsest grid.

$$S(q) = \int_a^b L(t, q(t), q'(t)) \quad dt \tag{3.17}$$

$$\frac{\partial L}{\partial q_i}(t, \overline{q(t)}, \overline{q'(t)}) - \frac{d}{dt}\frac{\partial L}{\partial q_i}(t, \overline{q(t)}, \overline{q'(t)}) = 0 \qquad for \quad i = 1, ..., n \tag{3.18}$$

After applying the minimization of the energy function of the image, we will obtain for each pixel a component in the x-direction and a component in the y-direction, thus obtaining an image with two channels per pixel. For the realization and application of the above described, the implementation in python, Optical flow method based Coarse2Fine warping method from Thomas Brox has been used. Below are some examples of the Optical flow computed with the script PyFlow[20]:

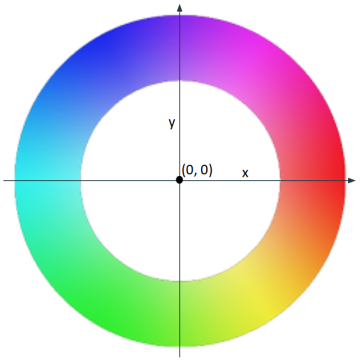


Figure 3.3: Optical Flow Color axis map

Figure 3.4: Left image is an example of two consecutives frames. Right image is the Optical Flow output between those images.

3.3 Exponential Moving Average

Exponential Moving Average (EMA) is a type of mean where the most recent samples have a greater weight. Similar to the Simple Moving Average (SMA) but with more sensitivity to the closest changes. It is computed using the following expression:

$$EMA_j^t = \alpha \cdot O_j^t + (1 - \alpha) \cdot EMA_j^{t-1} \tag{3.19}$$

$$EMA_j^0 = O_j^0 \tag{3.20}$$

The alpha parameter defines the weight of the current image with respect to the past, that is, the higher it is, the lower the importance of the previous images, and the smaller it is, the greater the importance. It should be emphasized that the importance of the past will have a decreasing exponential weight:

$$EMA_j^t = \alpha O_j^t + \alpha\beta O_j^{t-1} + \alpha\beta^2 O_j^{t-2} + \alpha\beta^3 O_j^{t-3} + \cdots \tag{3.21}$$

Therefore, this technique will be interesting as long as the previous information belongs to the same video and scene since otherwise, the past will not have relevance to generate the next image.

3.4 Shot Detection

In the same video, different scenes may appear, which must be detected in order to not take into account the information prior to that scene. Using the exponential moving average (EMA) methodology explained above, it is necessary to know where those changes occur so as not to generate the image taking into account frames that do not belong to the image in question. Let's suppose that, a car appears on the video moving along the road, and there is a change of scene, then the interior of the car is shown, in this case, the information of the moving vehicle from outside has no visual relationship with the object that is focusing inside the car. For this reason, when there is this type of change in the images, it is necessary to be aware of it. In addition, it is also necessary to define which motion image to calculate, since, the result of the difference between images has no meaning, and does not provide us with the movement of an object.

Thus, at the moment that there is a change of scene, the movement between those two images will be defined as a heatmap of value 0 in all the points, and information from the images generated previously will not be used. In the next frame, the FD will be calculated, but neither will the information of the previous frame be taken into account since we have defined it as 0.

Currently, there are several ways to be aware of the changes of scene in a video. We soon realize that it is a complex task: Let's suppose that a bird appears flying in the video where the camera also moves at the same speed, in this case, the background will be in continuous movement but the scene will be the same. Note that, in this case, we detect the movement of the background but not the main object. Then it will be understood that it is a complex task, and for this reason (it is not the objective of this work) a relatively simple system has been used to detect the typical and most relevant changes of the videos:

Given a couple of consecutive images, the proposed script will return True or False indicating whether it is a scene change or not. To make this binary decision, it is determined as follows: The 3 indices described below are calculated, which each of them returns True

or False and to finally decide the result is considered positive if at least two of these three indicators are positive:

- Sum of absolute differences (SAD): Consists of calculating the sum of all Euclidean distances between the color vector associated with each pixel and the same one of the previous frame. SAD is very sensitive to small changes in the same scene: rapid movements of the camera, explosions or simply turning on the light in a dark room can produce a false detection. On the other hand, SAD does not detect soft cuts very well. With this measure, you can measure the strong changes in scenes very well.

$$SAD_t = \sum_j ||x_j^t - x_j^{t+1}||_2 = \sum_j O_j^t \tag{3.22}$$

O_j^t : Heatmap resulting from the frame differencing, one component per pixel

$x_j^t = [r, g, b]^T$: Full image with 3 components per pixel

- Histogram difference (HD): Consists of calculating the difference between the histograms of each frame. To measure this distance Chi-Square is used because in general, it obtains better results compared with the Log-likelihood statistic or Histogram intersection [21]. HD is not as sensitive to small changes in the same scene as it is SAD and therefore produces fewer false detections. The main problem is that the two images can have the same histogram while the content is extremely different. For example, an image of a beach could have the same histogram as a desert with a blue sky. For the same reason, this indicator is not guaranteed to recognize sudden changes in the scene. The chi-square distance between two rows l, k is defined below:

$$HD(l,k)_t = \sum_j \sqrt{\frac{1}{x_{+j}} \cdot \left(\frac{x_{lj}}{x_{j+}} \cdot \frac{x_{kj}}{x_{k+}}\right)^2} \tag{3.23}$$

$$x_{i+} = \sum_j x_{ij} \quad , \qquad x_{+j} = \sum_j x_{ij}$$

$$M = \begin{bmatrix} x_{l,0}^t & x_{k,0}^{t-1} \\ x_{l,1}^t & x_{k,1}^{t-1} \\ \vdots & \vdots \\ x_{l,n}^t & x_{k,n}^{t-1} \end{bmatrix} : \text{Matrix of the two histograms}$$

- Edge change ratio (ECR): The ECR attempts to compare the actual content of two frames. Transforms both frames into edge images, (extracts the contours of the objects within the images). Then, compare these edge images using dilation to calculate the probability that the second frame contains the same objects as the first frame. This metric is very sensitive to sudden changes of scenes and at the same time can detect smooth cuts. The final calculation to determine the ECR of an image is shown below:

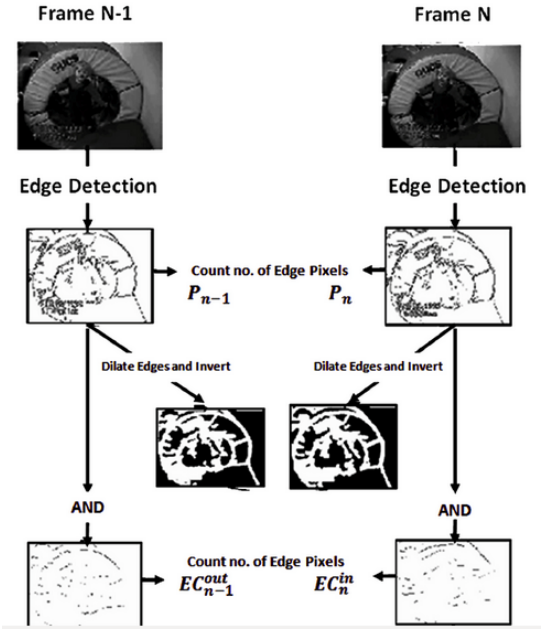


Figure 3.5: Steps of edge change ratio

$$ECR_t = max(ECR_t^{in}, ECR_{t-1}^{out}) = max\left(\frac{\sum EC_t^{in}}{\sum P_t}, \frac{\sum EC_t^{out}}{\sum P_{t-1}}\right) \tag{3.24}$$

ECR_t^{in} : Edge Change Ratio of the edges of the input edges

ECR_{t-1}^{out} : Edge Change Ratio of the points of the trailing edge

P_t : Total points of the edges

In the image above, an outline of the operation of the ECR calculation process is shown. At first, the edges of the two images (Canny edge detector) are detected. The number of total points respectively for each image is then calculated and stored in P_t and P_{t-1}. Then the dilation is applied and the images inverted. In order to detect the difference of points, the AND function is applied between the dilated and inverted image of the frame f_{n-1} and the image obtained with the detected edges of the image f_n. The same operation is performed with the two remaining images. The ECR is considered between two frames as the maximum between the entry points and the exit points.

The following image (Figure 3.6) shows the three indicators for a video section with about 600 frames in total (represented by the x-axis). All metrics are normalized between values [0, 1]. As we can see, there are 5 drastic changes of scene in this section of a video, where to a lesser or greater extent it is collected by the three indicators presented previously.

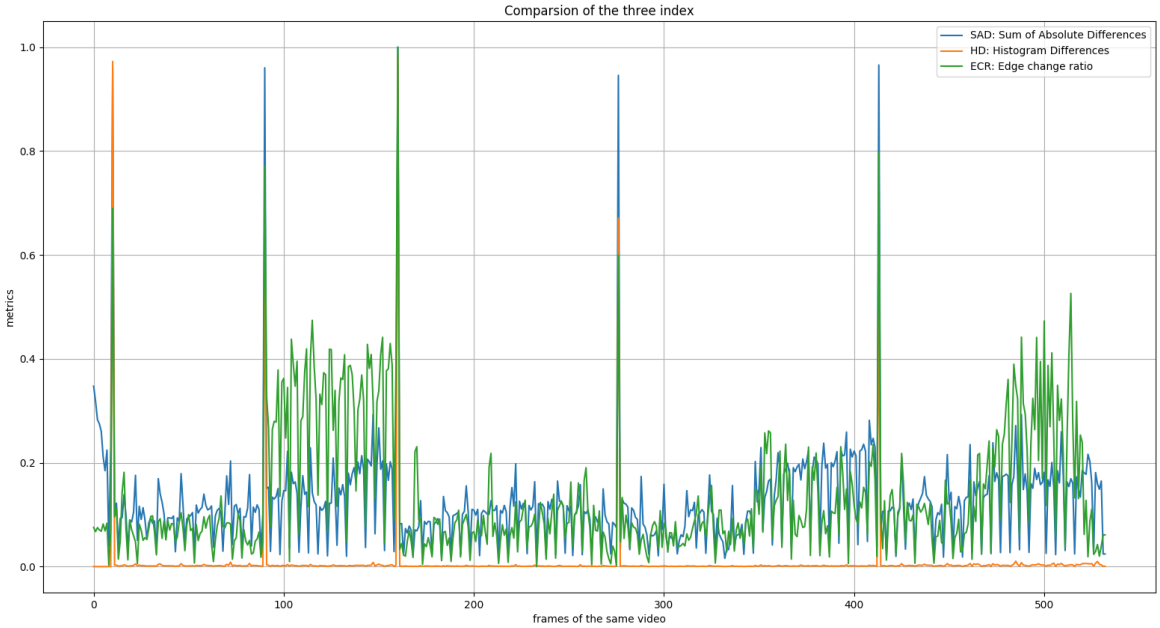


Figure 3.6: Steps of edge change ratio

Bearing in mind that it is necessary to detect when there is a drastic change in these indexes, the information of previous images will have to be stored in order to know when there is a significant relative maximum. In the presented script, an average of the 10 previous samples is calculated and when this exceeds x times the value of the average is considered a maximum. This threshold has been chosen so that it works well for the dataset in question (DHF1K). Note that for the scene changes less than 10 frames, this system will not work correctly.

3.5 Statistical Parameters

During the execution of this thesis, different statistical parameters have been used in order to extract information from the tests performed. Here is a brief summary of them:

- Variance is a measure of dispersion defined as the hope of the square of the deviation of the said variable with respect to its mean. It can be interpreted as a measure of the dispersion of the samples with respect to the mean.

$$\sigma_n^2 = E[X - \overline{X}^2] = E[X^2] - E[\overline{X}]^2 = \left(\frac{1}{n}\sum_{j=1}^{n} x_j^2\right) - \overline{X}^2 \tag{3.25}$$

n : Number of samples

x_j : Value of each sample

$\overline{x}$: Mean of the data

- Kurtosis is a measure of the "tailedness" of the probability distribution of a real-valued random variable. The standard measure of kurtosis, originating with Karl Pearson, is based on a scaled version of the fourth moment of the data or population. This number is related to the tails of the distribution, not its peak. We can define kurtosis in the following way:

$$E[x^p] = \frac{\sum_j^n x_j^p}{n} = \alpha_p \tag{3.26}$$

$$E[(x-m)^4] = \alpha_4 - 4\alpha\alpha_3 + 6\alpha^2\alpha_2 - 3\alpha^4 = \mu_4 \tag{3.27}$$

$$kurtosis = \beta_2 = \frac{\mu_4}{\sigma^4} \tag{3.28}$$

In the normal distribution, it is verified that $\mu_4 = 3\sigma^4$, where μ_4 is the moment of order 4 with respect to the mean and the standard deviation. Therefore, the following definition of the kurtosis coefficient is more widespread:

leptokurtic: $\beta_2 > 3 \quad and \quad g_2 > 0$: more pointed and with tails thicker than normal
platykurtic: $\beta_2 < 3 \quad and \quad g_2 < 0$: less pointed and with tails thicker than normal
mesokurtic: $\beta_2 = 3 \quad and \quad g_2 = 0$: when it has a normal distribution

$$g_2 = \frac{\mu_4}{\sigma^4} - 3 \tag{3.29}$$

Chapter 4

Experiments

Summary

The work done as well as the experiments and their respective results are presented below. The results obtained and their direct implications on the initial hypothesis are discussed. The process of obtaining the motion of the videos is explained in detail and how they have been integrated together with a static model. The own model is also presented to realize said mixed implementation. It is intended to give a detailed explanation of the entire process to establish the basis for the final conclusions.

4.1 Obtaining the movement

At first, the frame differencing has been computed, which consists of measuring the vector distance between two frames, as well as OpticalFlow. In the following sections, I will explain in depth the difference between these two methodologies. We will focus only on the model based on the Frame Differencing (see in 3.1):

Figure 4.1: examples of the frame differential heatmap

The hypothesis consists of verifying that movement is important in the generation of saliency maps in the video. To begin to verify this idea, as explained before, the saliency maps were generated, that is, they were created only and only from the movement predicted by this technique. From these maps, the metrics previously described for each frame of the validation dataset were computed. As expected, it can be seen (Table 4.1) that the Frame Differencing obtained a lower average score than SalBCE (Table 4.1). However, that does not mean that it does not make a good prediction, it must be borne

in mind that this system does not capture static objects without movement. For that reason, all those videos where still objects appear, or moments of the video where there is no motion will not be detected.

The rate between the frames where FD obtained better results with respect to all the frames has been computed. The obtained results show that there is an improvement between ∼12% and ∼24% of all frames, that is to say, that almost in 1/4 of the frames, the simple technique of FD showed a better score (Table 4.1, row 14). At first, it was expected that this technique would obtain better results depending on the type of video, and according to the results obtained (see Figure 4.2) where the result is shown in orange using a model without taking into account motion, and in blue the saliency maps with the FD technique, you can see that in almost any video is getting better average results from all the frames in each video.

-	AUC-Judd	SIM	S-AUC	CC	NSS
SalBCE	0.8909	0.2667	0.7102	0.3819	2.135
Frame Differencing	0.7052	0.1534	0.5917	0.1335	0.7493
Rate of FD doing it better than SalBCE	12.33%	14.98%	23.81%	13.17%	13.48%

Table 4.1: Experiment results

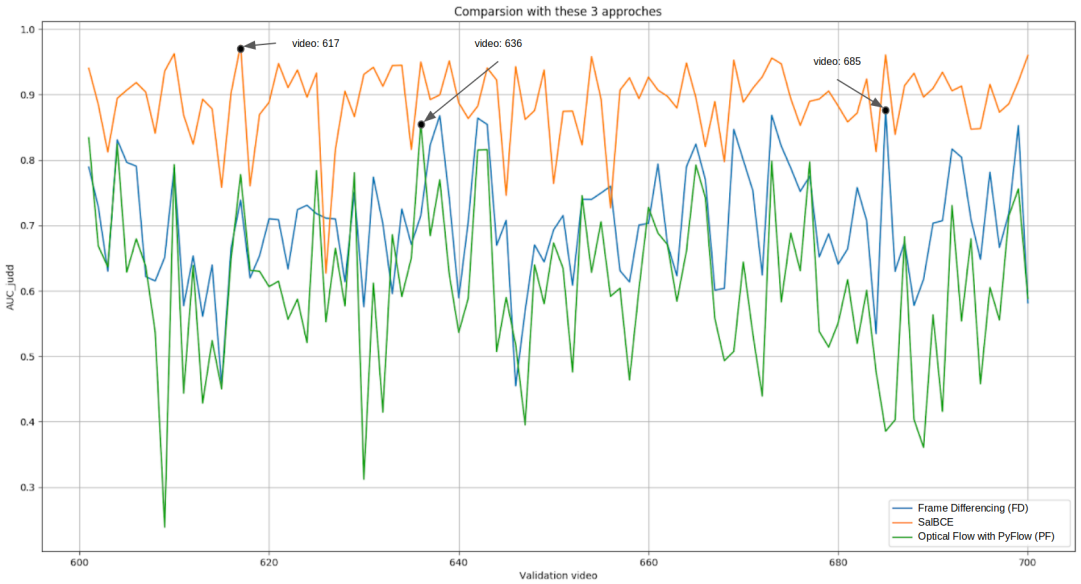


Figure 4.2: Comparison of results in the dataset validation

To generate the following graph (Figure 4.3), the difference in the value of the metrics for each frame between the dynamic model and the static model has been computed. All those points/frames that obtain a positive value means that they have obtained a better result while, if a negative value is obtained, the static model has obtained a better score. In addition, the absolute value of that difference shows the magnitude of the distance. In blue, the frames are shown where a positive value has been obtained, and in orange, a negative value. It can be verified that, although in almost any video it has obtained better results, there is a great number of frames that it has. Occasionally there are certain videos where you get a much higher score and others that lower, but it is found that the improvement exists in certain frames of all videos:

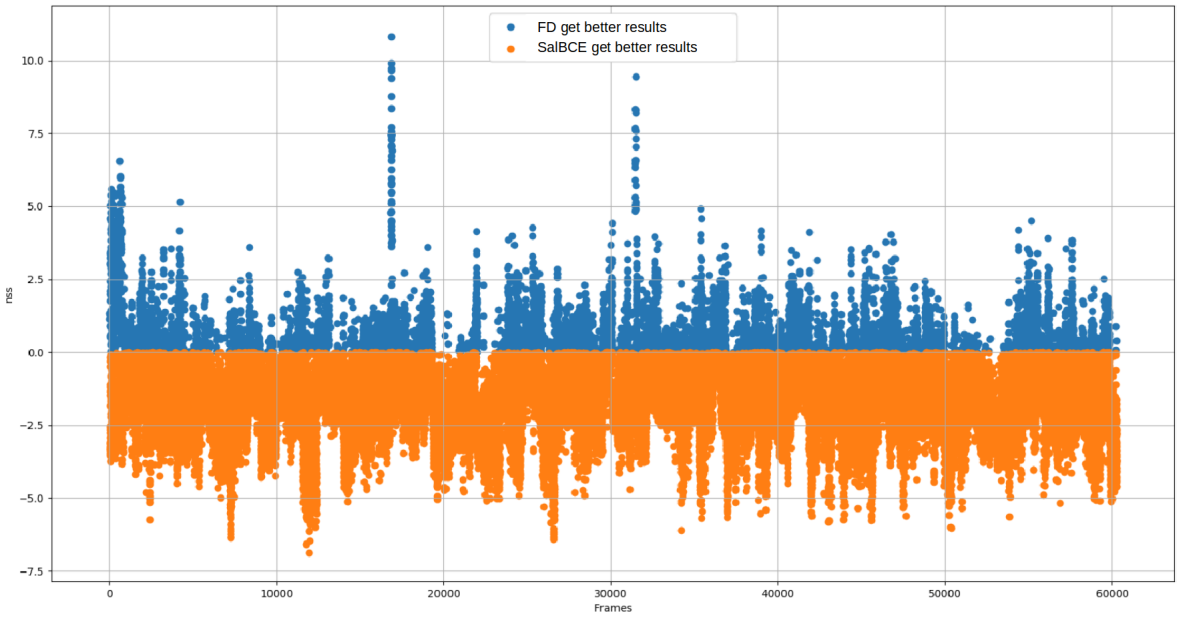


Figure 4.3: Comparison of results in the dataset validation

Even so, the average is still lower than the static model (Figure 4.4), and can be checked by comparing the metrics of all the frames of the same video. Next, the value of the static model is shown in orange, and without contemplating movement in blue (dynamic model). You can see that although in most frames the result is much worse, there is a certain number of frames which is higher:

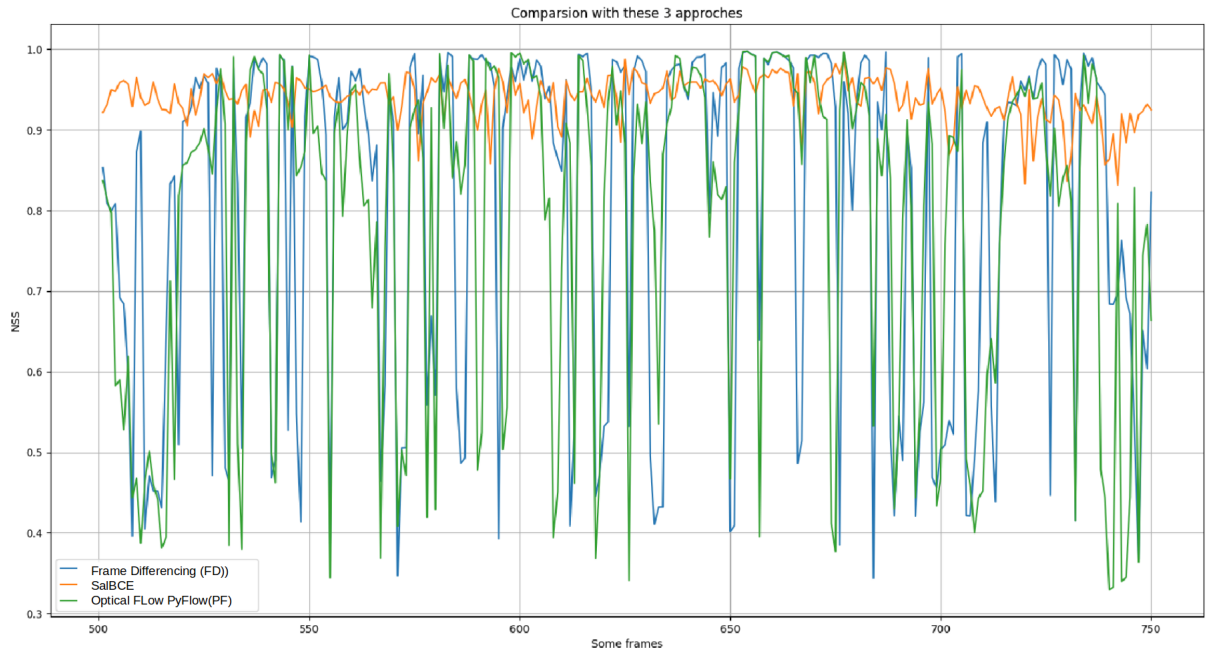


Figure 4.4: Comparison of the results of a given video of the validation dataset

The trivial conclusion that is obtained from these results, is the fact that this system only captures information when the object which we set is in motion, for this same reason, the only improvement is captured in those frames where this dynamism exists. Those in which there is no motion or is composed of not only dynamic parts, a worse result will be obtained.

4.2 Motion integration with static objects

In a first approximation, it has been proposed to generate a new mixed prediction, that is, from the two maps generated by each system, use both to generate a map that integrates the two types of information. To generate such predictions, two systems have been proposed:

- Mixed integration: Apply different mathematical functions to generate another map from the two, such as making a mean, or taking the maximum of each map, etc ..

- Binary Classifier: Based on other statistical parameters, which do not include the ground truth, decide in each prediction which system will get the best score and which one, one or the other.

Mixed integration

In the first methodology, 4 basic functions are proposed to generate the saliency maps. In this case, no statistical parameter has been applied. After the application of these functions all the saliency maps have been norm by range between (0, 255):

- Parameter α: A weighted sum has been applied to the two resulting maps.

$$map_{new} = \alpha \cdot map_{static} + (1 - \alpha) \cdot map_{dinamic}$$

- Sum: The direct addition pixel to pixel of the two saliency maps has been applied.

$$map_{new} = map_{static} + map_{dinamic}$$

- Maximum: The maximum pixel to pixel function has been applied to both maps.

$$map^i_{new} = max\{map^i_{static}, \quad map^i_{dinamic}\}$$

- Minimum: The minimum pixel to pixel function has been applied to both maps.

$$map^i_{new} = min\{map^i_{static}, \quad map^i_{dinamic}\}$$

In Table 4.2 it can be verified how the results have been similar, although the one corresponding to the weighted average with $\alpha = 0.5$, has been the one that has obtained the best results in all the metrics studied. Even so, the results are not better with respect to SalBCE except in SIM and S-AUC metric.

-	AUC-Judd	SIM	S-AUC	CC	NSS
$\alpha = 0.5$	0.8730	**0.3412**	**0.7127**	0.3412	1.906
addition	0.8730	0.2289	**0.7127**	0.3412	1.906
maximum	0.8640	0.2192	0.7091	0.3238	1.813
minimum	0.7813	0.2153	0.6226	0.2395	1.323
SalBCE	**0.8909**	0.2667	0.7102	**0.3819**	**2.135**
Frame Differencing	0.7052	0.1534	0.5917	0.1335	0.7493

Table 4.2: Mixed integration results

Note that although α parameter and addition seem exactly the same, there is a difference. While the first one is computing the average of both maps and its value never is more than 255, in the second one, before normalizing we truncate the saliency map. So, all the pixels where the sum of both images in that pixel is greater than 255 will be represented as the same value.

Binary Classifier

In the second methodology, the use of statistical parameters is proposed in order to predict which model will generate a better prediction. The kurtosis, the variance and the amount of movement have been used to try to make this prediction. The previous statistical parameters have been computed on the FD. The main idea is to decide without having knowledge of the ground truth which of the models will obtain the best result and choose one or the other as appropriate.

Next, the difference in a video between the dynamic model and the static model is shown in blue. In orange, the total amount of movement is shown in the first case (Figure 4.5), and in the second, is shown the kurtosis (Figure 4.6).

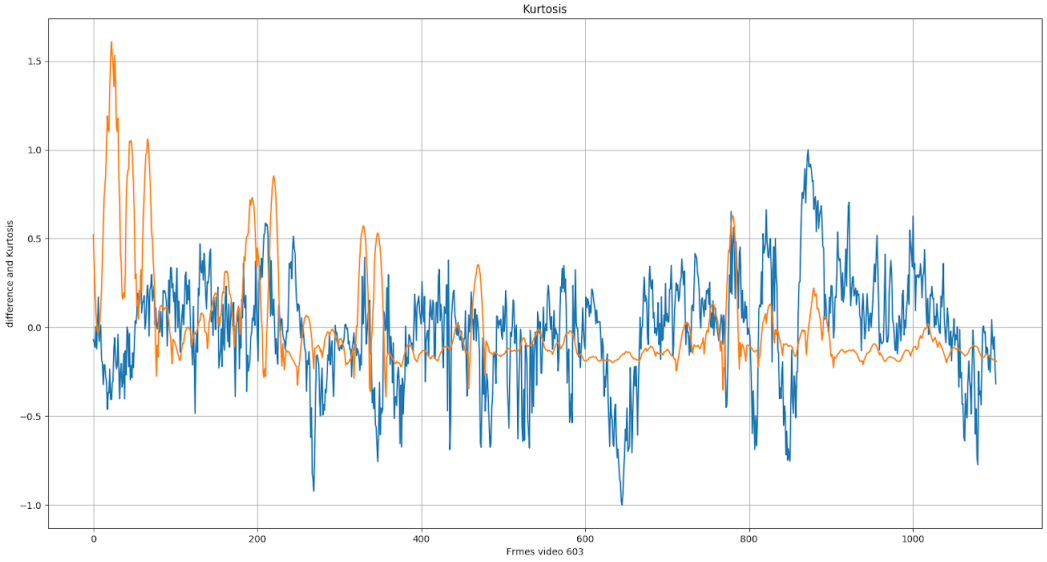


Figure 4.5: Comparison of kurtosis with the good results

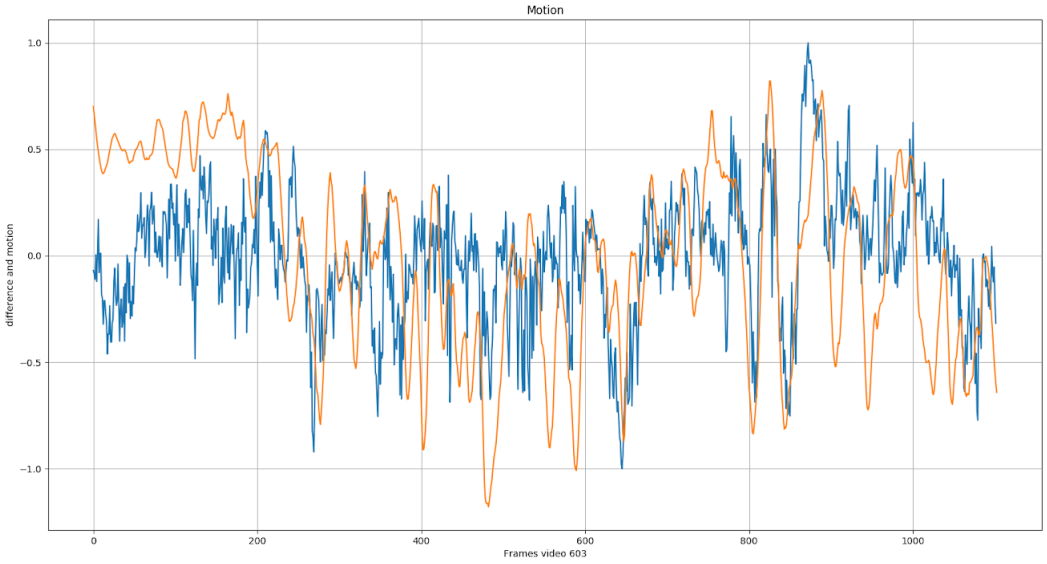


Figure 4.6: Comparison of motion with the good results

It can be verified that there is a certain relationship between the amount of movement and a better implementation with the model that implements the movement. Actually, the most important thing is not the numerical value, rather if it is positive or not. To do

this, the function $sign\{kurtosis, motion, variance\}$has been computed and it has been proved that an important correlation exists.

In order to generate the final saliency maps, different implementations have been proposed:

- Choose the dynamic model only those frames where the variance of the FD associated with that frame is less than -0.5. Table 4.3

- Choose the dynamic model only those frames where the amount of movement of the FD associated with that frame is greater than 0.5. Table 4.3

- Choose the dynamic model only those frames where the kurtosis of the FD associated with that frame is greater than 0. Table 4.3

- Choose the dynamic model only those frames where the AND function is fulfilled between the amount of movement, the variance and the kurtosis of the FD associated with that frame is greater than 0. Table 4.3

- Choose the dynamic model only those frames where the AND function is fulfilled between the amount of movement is greater than 0.5, the variance less than -0.5 and the kurtosis higher than -0.5 of the FD associated with that frame. Table 4.3

-	AUC-Judd	SIM	S-AUC	CC	NSS
variance from (FD, < -0.5)	0.8687	0.2513	0.6955	0.3494	1.952
average motion (from FD, > 0.5)	0.8684	0.2531	0.6986	0.3541	1.981
kurtosi (from FD, > 0)	0.8237	0.2261	0.6636	0.2883	1.614
mix ($\alpha > 0$)	0.8788	0.2602	0.7046	0.3671	2.063
mix ($> 0.5, > 0.5, > -0.5$)	**0.8946**	**0.2698**	**0.7147**	**0.3876**	**2.176**
SalBCE	0.8909	0.2667	0.7102	0.3819	2.135
Frame Differencing	0.7052	0.1534	0.5917	0.1335	0.7493

Table 4.3: Binary decision results

It can be seen that all the models have obtained a similar score and lower than the static model. However, the last implementation (Table 4.3), has obtained slightly higher results. It must be taken into account that the decision of the parameters is not optimized to obtain the best results. The difference is very small so it could be the case that in fact due to variations in the computation of the metrics there would not be a big difference between one model or another, so we have to improve our model to include motion in video saliency prediction.

In these experiments, we can conclude that motion is important. Although a solid solution has not been proposed as far as implementation is concerned, the foundations of its necessary implementation in prediction have been proposed, as well as a corroboration of the importance of motion in saliency prediction task.

4.3 Optical Flow implementation

Once the involvement of the movement in the prediction of salience has been demonstrated, it has been decided to improve the capture of this movement. PyFlow (see in 3.2 has been used in order to obtain the Optical Flow of the dataset. Unlike the differencing frame, the optical flow identifies the direction and magnitude of that movement. To be able to represent this direction, it is necessary to map the addresses (x, y) with the spectrum (R, G, B) in the range of (0, 255). Although it is also possible to calculate the magnitude, it is preferred to use the extra information provided by this model.

For this study we have used the implementation made in C ++ instead of FlowNet 2.0[22], although with this second method, we could probably obtain higher speed of generation of the Optical Flow and greater precision in the detection of this.

In Figure 4.2, the different values obtained with SalGAN (orange), FD (blue) and Optical Flow (green) are compared in the AUC_judd metric. It can be seen that this system does not obtain a better average score in the videos. It has similar behavior to the FD although in some videos they obtain better results, nevertheless, it obtains a lower percentage of frames better than the static model, as well as a worse score in all the metrics (Table 4.4). In the figure some frames it can be seen that similar behavior is obtained in FD. An improvement in certain frames of the same video, obtaining in the rest a result much lower than the one obtained with SalBCE.

On the other hand, it has been verified what is the similarity percentage between these two approximations, that is, what percentage of the frames that best obtain the FD also obtained it in the Optical Flow, and surprisingly the result was approximately $\sim 60\%$. It means that $\sim 40\%$ of frames, which scored better, are different frames. So if we take into account the extra percentage, the percentage of better frames is increased, using only the technology with movement:

-	AUC-Judd	SIM	S-AUC	CC	NSS
Rate of FD doing it better than SalBCE	12.33%	14.98%	23.81%	13.17%	13.48%
Rate of PF doing it better than SalBCE	12.09%	14.56%	23.41%	12.14%	12.55%
Rate of PF doing it better than SalGAN in which FD did	52.9%	61,7%	52.8%	54.2%	61.1%
Rate of FD and PF doing it better than SalBCE	18.02%	20.56%	34.86%	18.73%	18.36%

Table 4.4: Ratio of improvement, of movement respect without movement

It can be observed in the previous table that with a simple movement model, that is, decide that where there is movement there is salience, it is achieved in the best case, an improvement of 34.86% of the frames, obtaining a clear correlation between movement and salience.

To understand what properties a video must have in order to see which model works best, it has been decided to choose certain videos. Specifically, the visualization of videos 626, 655, and 666 has been chosen for the study, coinciding in the peaks where the best result is obtained (see Figure 4.7), with both movement systems and SalBCE where it obtains a worse result. In these videos is where motion models get a better percentage of frames than the static model:

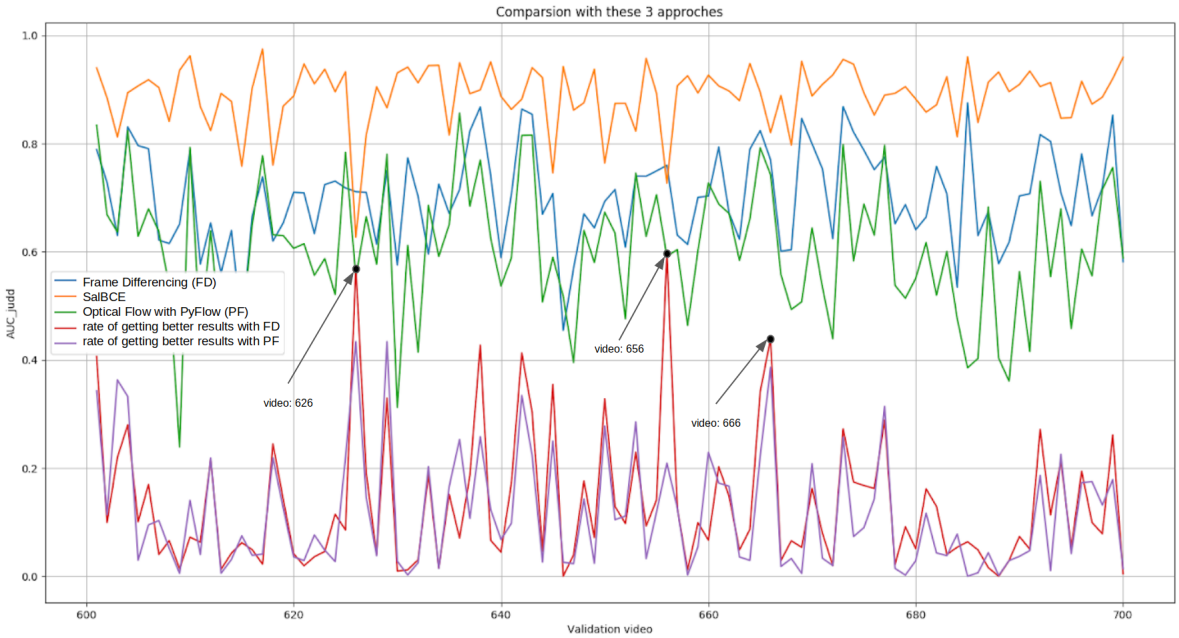
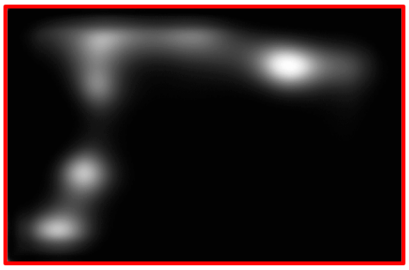


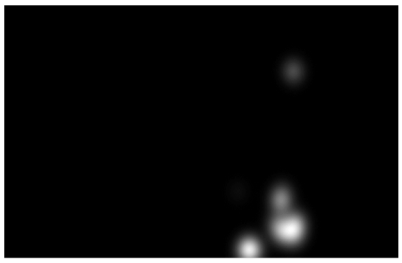
Figure 4.7: Comparison of results in the dataset validation and the rate

Video 626: It can be observed that the movement occurs in the golf ball located in the lower right part of the image. You can check in the prediction made with SalGAN that at no time does the ball appear. However, in the two systems with the movement they have only marked that point, obtaining a greater result, with a kinship to the ground truth much higher.

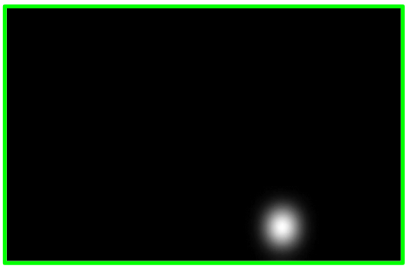
DHF1K video (video 626, frames 0-200)



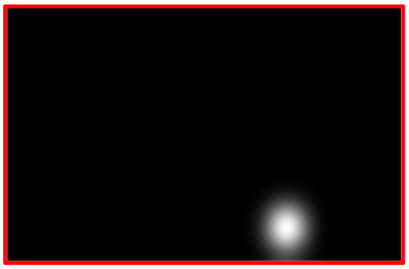
SalBCE (video 626, frames 0-200)



Ground Truth (video 626, frames 0-200)



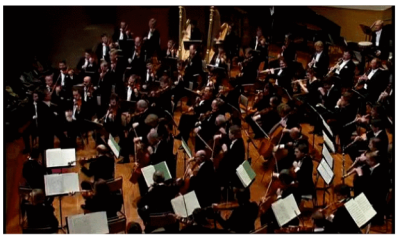
Frame Differencing(video 626, frames 0-200)



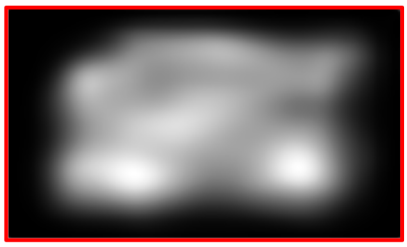
Optical Flow with PyFlow (video 626, frames 0-200)

Figure 4.8: Video 626: DHF1K video, ground truth, PF and FD

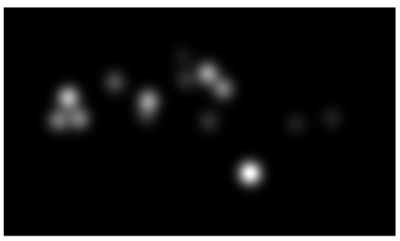
Video 656: An orchestra can be observed playing the instruments, where there is a lot of movement but in small parts inside the video. In the static model, a salience is observed in almost the entire frame, however, with FD it is able to obtain better this information, obtaining a better result again.



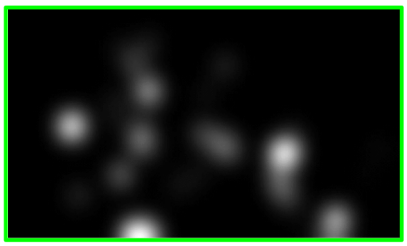
DHF1K video (video 656, frames 0-200)



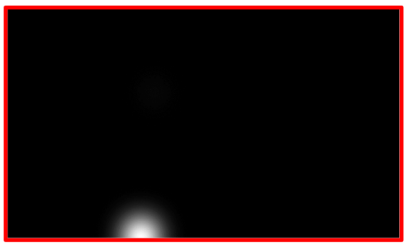
SalBCE (video 656, frames 0-200)



Ground Truth (video 656, frames 0-200)



Frame Differencing(video 656, frames 0-200)



Optical Flow with PyFlow (video 656, frames 0-200)

Figure 4.9: Video 656: DHF1K video, ground truth, PF and FD

Video 666: In the video, you can see a woman moving around the kitchen. In the static model, it is mainly shown the fixed objects of the countertop beside the girl. However, in dynamic models, they better capture the movement of the girls predicting in a better way.

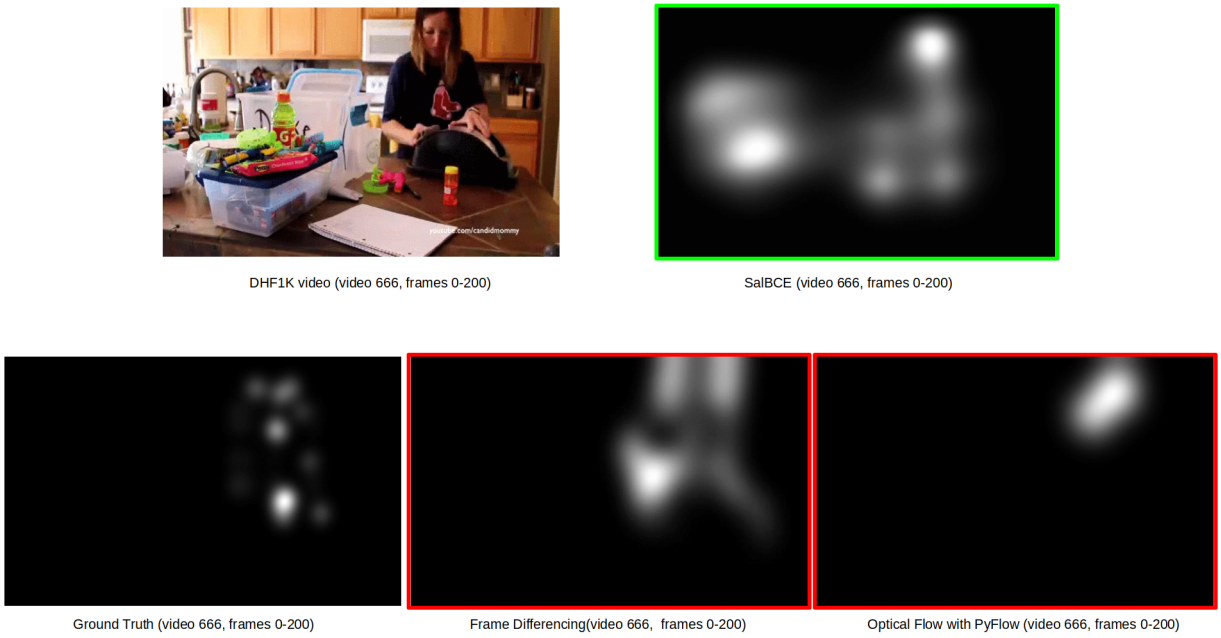


Figure 4.10: Video 666: DHF1K video, ground truth, PF and FD

It can be concluded that in those videos where scenes with movement appear, and these are the most relevant points from the point of view of salience, dynamic systems tend to obtain better results. However, the optimal point would be the inclusion of both information in the same model in order to get a better prediction.

4.4 Motion with neural networks approach

The model proposed in the section 2.1.4 based on Convolutional Neural Networks (CNN) has been used to generate the saliency maps implementing the information extracted from the original images and the motion of said images. In order to improve their metrics values, we have made the inference from the SalBCE model in the frame differencing images, optical flow images and the image with a mix of both techniques.

In Figure 4.11 it can be seen how the model using the complete image (FD and PF as in one stream method [11]) with all the information related to the motion, obtain a better approximation. In the NSS and SIM metrics is where you can see more difference and greater improvement:

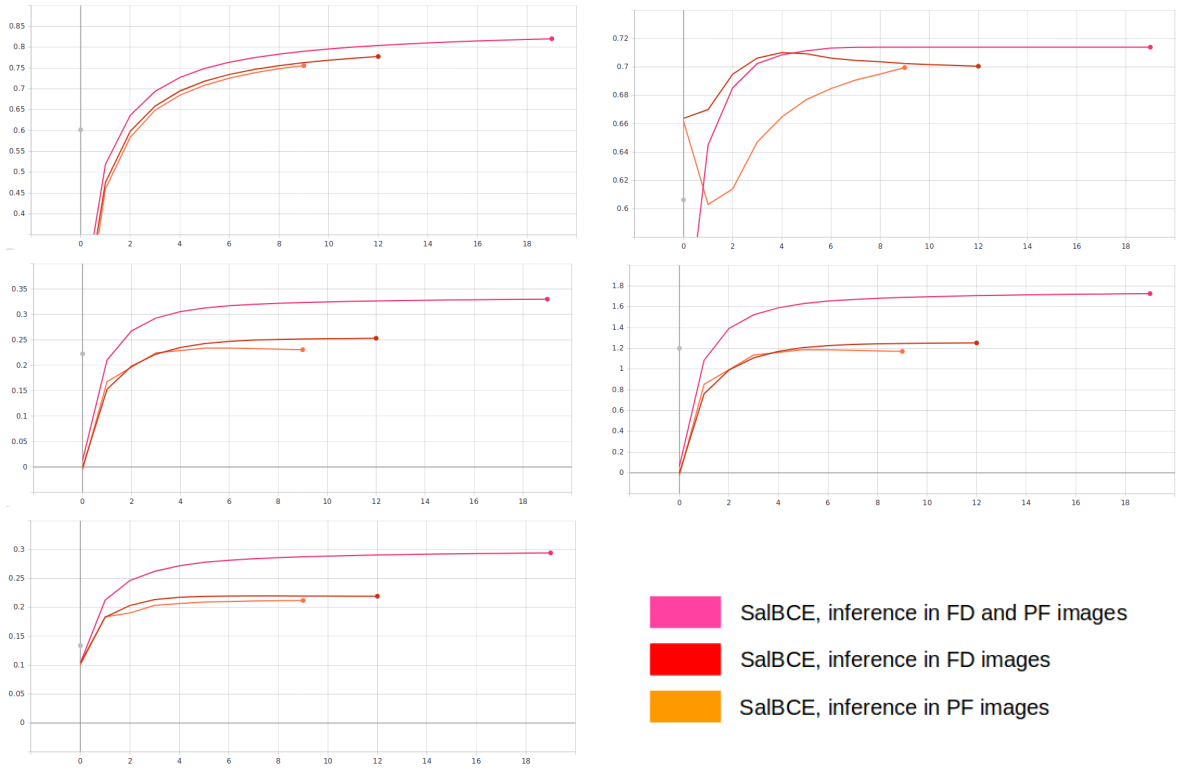


Figure 4.11: Metrics results of training SalBCE. The inputs are the three different types of motion images. The x-axis represents the epochs and in the y-axis the value of each metric. Top left: AUC_Judd, Top right: AUC_shuff, Middle left: CC, Middle right: NSS, Bottom left: SIM

SalBCE inference in:	AUC-Judd	SIM	S-AUC	CC	NSS
FD	0.778	0.219	0.700	0.253	1.25
PF	0.756	0.212	0.700	0.231	1.17
FD and PF	0.850	0.295	**0.716**	0.335	1.78
RGB	**0.873**	**0.342**	0.645	**0.408**	**2.23**

Table 4.5: Metrics results making inference in the motion generated images

In the table 4.5 it can be checked how the models exclusively using the movement information is not enough to overcome the SalBCE metrics. It must be taken into account that due to the lack of the initial wights of SalBCE because they are not published, it has been necessary to re-train the model, reason why the values of SalBCE in the RGB images differ a little with respect to the offers during the work. (At first, only the images from validation dataset were generated and to train the complete model it was necessary to replicate the conditions of SalBCE training, and once we got the right weights to make inference in de validation as well as the training dataset

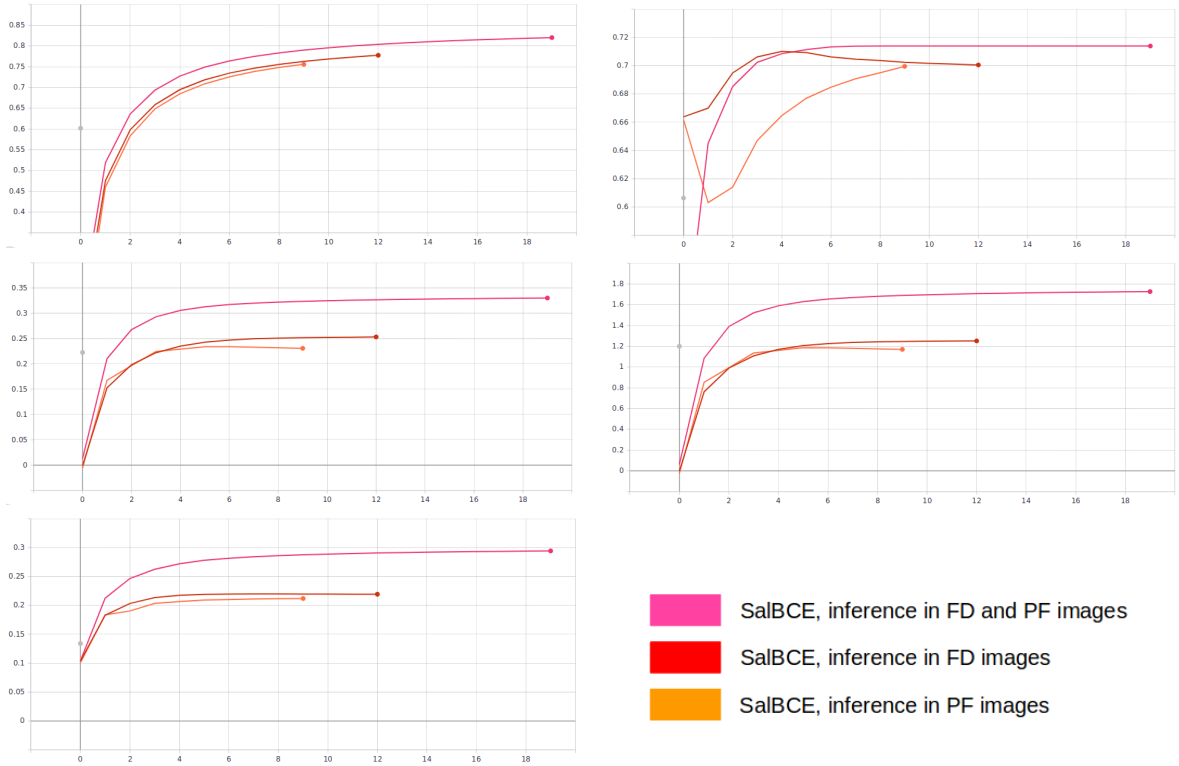


Figure 4.12: Metrics results of training the last architecture to mix both saliency maps.(from RGB and motion). Top left: AUC_Judd, Top right: AUC_shuff, Middle left: CC, Middle right: NSS, Bottom left: SIM

In Figure 4.12 are the results of training the custom CNN head architecture proposed to generate the final saliency map from the two corresponding outputs of making an inference with SalBCE in the RGB images and those of the motion. There have been done 20 epochs with a descendant learning rate.

SalBCE inference in:	AUC-Judd	SIM	S-AUC	CC	NSS
RGB	0.873	0.342	0.645	0.408	2.23
FD and PF	0.850	0.295	0.716	0.335	1.78
RGB, FD, and PF	**0.874**	**0.345**	**0.750**	**0.410**	**2.27**

Table 4.6: Metrics results making inference in the motion generated images

In Table 4.6 it can be seen how the final maps generated are quite similar to the ones without taking motion into account, except in the SIM metric, all the others have been improved in respect to the initial ones. Although the improvement is relatively small and an impressively high absolute result has not been obtained comparatively with other implementations (see in Table 4.13), the metrics of the initial system have been improved, allowing to say at least the inclusion of the movement has certain relevance.

The structure described in the section 2.1.4 has also been used in order to simplify the other models in the section 2.1.4.1. Unlike the latter model, the results have been very similar with all the tests carried out. The static information has been combined with different movement information: (RGB + FD, RGB + OF, RGB + FD + OF, RGB). Little improvement has been obtained in any case. With this result we can conclude that it is necessary to use another type of architecture to better implement the dynamic information.

Below is the comparative table of the different approaches to solve the saliency prediction. We can get an idea of the values obtained in this work comparatively with the first positions. It can be proven that although the best results are not obtained, it is enough to prove the importance of the movement:

Method	AUC-J	SIM	s-AUC	CC	NSS	Implement.	Size (MB)	Time (s)	DLM	D/S
ACLNet	0.890	0.315	0.601	0.434	2.354	Tensorflow	250	0.02	√	D
SalGAN	0.866	0.262	0.709	0.370	2.043	Theano	130	0.02	√	S
DVA	0.860	0.262	0.595	0.358	2.013	Caffe	96	0.1	√	S
SALICON	0.857	0.232	0.590	0.327	1.901	Caffe	117	0.5	√	S
OM-CNN	0.856	0.256	0.583	0.344	1.911	Tensorflow	344	0.05	√	D
Deep-Net	0.855	0.201	0.592	0.331	1.775	Caffe	103	0.08	√	S
Two-stream	0.834	0.197	0.581	0.325	1.632	Caffe	315	20	√	D
Shallow-Net	0.833	0.182	0.529	0.295	1.509	Theano	2500	0.1	√	S
GBVS	0.828	0.186	0.554	0.283	1.474	C		2.7		S
Fang et al.	0.819	0.198	0.537	0.273	1.539	Matlab		147		D

Figure 4.13: DHF1K video saliency leaderboard. https://mmcheng.net/videosal/. Consulted time: 18 May 2019.

Chapter 5

Environment

In order to work correctly, organized and safely, a configuration scheme based on the GitHub platform for code storage and a docker[23] for the correct development of this has been proposed. To perform code tasks, visual studio code has been used, together with an external laptop to the server where, through the ssh and scp protocols, they have been communicated.

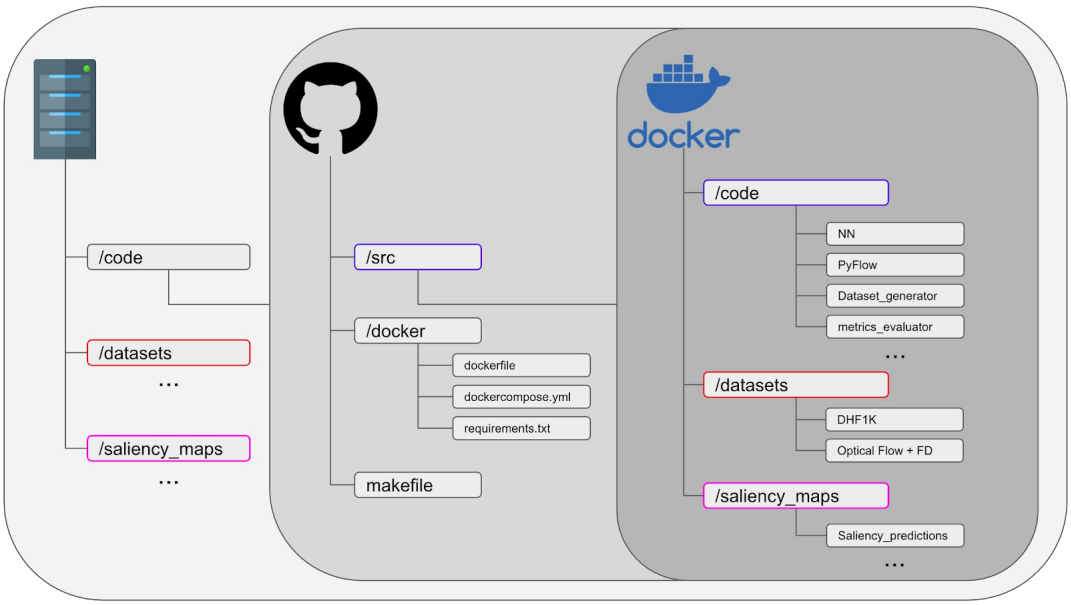


Figure 5.1: Enviorment

Due to the need for computing power and storage space, a server with the right components has been used (laptops do not have enough features for this kind of task). Thus, communication with the server has been carried out by means of a laptop. In order to

avoid compatibility issues and different versions, the docker technology has been used, so before uploading the programs to the server, local tests were possible (very easy for scalability). All the code has been periodically uploaded to the GitHub repository in order to maintain version control and have all the work saved at all times.

The difference of colors between folders serves to understand their connection between them, that is, the datasets folders and saliency maps (due to a large amount of information they store) have not been uploaded to GitHub, and therefore their files inside the docker it is a mapping of the folders inside the server.

Chapter 6

Budget

For the realization of the thesis a work space has been used at the University of Dublin (DCU) with sufficient material (Office, screen, servers, electricity, etc ...) to carry out the necessary tasks. Insight Center for Data Analitycs has provided my tutor Kevin McGuinness in conjunction with Xavier Giro-i-Nieto (From Universitat Politecnica de Catalunya) to oversee the thesis. The software used are open-source. However, in order to perform the necessary computation, it has been compared with cloud services:

- At AWS (Amazon Web Services)[1] similar GPU used as an Insight, it would cost 0.9$ + IVA per hour. If the use was prolonged (minimum of 3 years) the price would be 0.425$ + IVA.
- At GCP (Google Cloud Platform)[2] a similar GPU with the same amount of RAM it would cost 0.95$ + IVA. If a preemptile system is used (permission is given to google so that it can paralyze the processes up to 24h when they need it) the price is only 0.135$ + IVA.

	Amount	Wage/Hour	Dedication	Weeks	Total
Undergraduate researcher	1	10 $	35 h/week	15	5,250 $
Senior Engineer	2	35 $	1 h/week	15	1,050 $
GPU Tesla K80	1	0.95 $	80 h/week	15	1,140 $
Total					7,440 $

Table 6.1: Costs of total hours of work. Social Charges and taxes are not included

[1] https://aws.amazon.com/es/ec2/instance-types/p2/
[2] https://cloud.google.com/compute/pricing#sustained_use

Chapter 7

Ethics

In recent years, research in the field of computer science and artificial intelligence has grown exponentially. Largely due to the increasing computing capacity that this type of architectures requires, however, there is still a long way to go. With these new methodologies have begun to imagine concepts that were previously unthinkable, as could be the autonomous car or the prediction of diseases just by having a photo. The amount of possibilities that these technologies can offer is not even present in our lives. Going a step further, can we imagine a model where, for example, justice is using artificial intelligence to predict whether or not we will be guilty, even before committing any crime.

We are not really aware of the repercussion that can be predicting the fixations of the eye in images or videos. To what extent we can understand it as part of our privacy, we could ethically pass through a barrier. On the other hand, there are a lot of applications with which the saliency prediction could work very well. Apart from other tasks in artificial intelligence that this methodology could provide, it could currently be used in marketing or for example in video compression (maybe it is not necessary to transmit all the information of the whole image equally if, in the end, we only look at a part of that frame).

During the realization of the thesis, it has also been possible to identify certain bias depending on the type of video. The same action performed by different people, in gender, race, or even religion have different ground truth. This can be seen very well in sports videos carried out by a man or a woman. We understand then that there is a

variation of the ground truth in the function of certain parameters of the group that made these images. Taking a small group of people (see in Table 2.2) it is very likely that the result will be diverted towards a certain trend. The saliency maps not only depend on the images that we are seeing but also the people who are behind based on their memories and experiences. It could be interesting to use a group of children to see how their fixation maps are.

Chapter 8

Conclusion and Future Developement

During this thesis, I have learned a lot of concepts related to artificial intelligence, from the basic operation of a simple neural network to the Depp Neural Networks (DNN) as the one of SalBCE. It has also been very useful to start knowing the pythorch framework and implement my own architecture from scratch. With this project we provide the following contributions:

- Different approaches have been presented to demonstrate that motion is important in saliency prediction. Different graphs and findings have been shown using the DHF1K dataset.
- We have studied a state-of-the-art saliency model for static images named SalGAN and for videos, SalBCE. We have evaluated its performance in the DHF1K benchmark, rating second in the public leaderboard, under ACLNet.
- A custom CNN head has been presented using the well-known SalBCE to implement the motion in the prediction process. The bases have been laid to begin to develop a model that implements this concept.

The main goal has been accomplished, as shown in section 1.1, all the proposed objectives have been met, although with some nuance. It has been proved that, indeed, we tend to look more to the parts of the objects in motion. On the other hand, the extraction of the dynamic information has been carried out by means of two known techniques, FD and PyFlow (Lukas Kanade implementation), thus generating (as in the one stream methodology) an image with movement information. An implementation based on SalBCE has also

been proposed to integrate that movement in the calculation of saliency prediction. The results obtained with this architecture have not been as expected, although the difference has been small, we have obtained better results in all the metrics. In AUC-Shuffle, up to a 16.3 % improvement has been achieved with respect to the baseline BCE, however, in the other metrics, not much more than 1% improvement has been achieved.
Given the previous results, below are different ways to improve the results:

- To improve the obtaining of the movement, the use of other neural networks is proposed with the purpose of decreasing the computing time. During the execution of the work we have been studying the implementation of FlowNet2.0 [22] or more recently the use of PWC-Net [24] (is 17 times smaller in size, 2 times faster in inference, and 11 % more accurate on Sintel final than the recent FlowNet2 model). At the end of the work, a new implementation superior to all demands has been published, called SelFlow [25].

- The inclusion of a new more complex architecture that can be implemented - as in ACLNet - attention modules or LSTM between the two image channels, as well as an improved architecture for the merging of the groups of images.

- Train the model with different datasets, such as SALICON or UFC Sports, in order to obtain a greater variety of content and improve the accuracy of the prediction.

Although different new ways to improve movement are proposed, we believe that the best way to improve this system is to directly include the application of the first point, where the use of a new more precise architecture is necessary. At first, the results obtained with FD and OF were very positive, however, those calculated using this Custom CNN head have not been sufficient compared to the computation cost that is required. A system has been used where the movement information has been considered as important as the static information, and maybe it would be smarter to use this information to help the SalBCE in RGB to fix more in those areas with movement.

References

[1] I. Goodfellow, Y. Bengio, and A. Courville, "Deep learning. book in preparation for mit press," *URL¡ http://www. deeplearningbook. org*, 2016. 1

[2] W. Wang, J. Shen, F. Guo, M.-M. Cheng, and A. Borji, "Revisiting video saliency: A large-scale benchmark and a new model," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 4894–4903, 2018. 1, 10

[3] J. J. Nieto Salas, "Video saliency prediction with deep neural networks," B.S. thesis, Universitat Politècnica de Catalunya, 2019. 1, 9

[4] J. Pan, C. C. Ferrer, K. McGuinness, N. E. O'Connor, J. Torres, E. Sayrol, and X. Giro-i Nieto, "Salgan: Visual saliency prediction with generative adversarial networks," *arXiv preprint arXiv:1701.01081*, 2017. 8

[5] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv preprint arXiv:1409.1556*, 2014. 8

[6] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014. 9

[7] M. Jiang, S. Huang, J. Duan, and Q. Zhao, "Salicon: Saliency in context," in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2015. 9

[8] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick, "Microsoft coco: Common objects in context," in *European conference on computer vision*, pp. 740–755, Springer, 2014. 9

[9] M. Marszałek, I. Laptev, and C. Schmid, "Actions in context," in *CVPR 2009-IEEE Conference on Computer Vision & Pattern Recognition*, pp. 2929–2936, IEEE Computer Society, 2009. 10

[10] M. Sullivan and M. Shah, "Action mach: Maximum average correlation height filter for action recognition," in *CVPR*, pp. 1–8, 2008. 10

[11] A. Khoreva, R. Benenson, E. Ilg, T. Brox, and B. Schiele, "Lucid data dreaming for video object segmentation," *International Journal of Computer Vision*, pp. 1–23, 2018. 11, 50

[12] M. Kummerer, T. S. Wallis, and M. Bethge, "Saliency benchmarking made easy: Separating models, maps and metrics," in *Proceedings of the European Conference on Computer Vision (ECCV)*, pp. 770–787, 2018. 14, 18

[13] Z. Bylinskii, T. Judd, A. Oliva, A. Torralba, and F. Durand, "What do different evaluation metrics tell us about saliency models?," *IEEE transactions on pattern analysis and machine intelligence*, vol. 41, no. 3, pp. 740–757, 2019. 14

[14] S. N. Tamgade and V. R. Bora, "Motion vector estimation of video image by pyramidal implementation of lucas kanade optical flow," in *2009 Second International Conference on Emerging Trends in Engineering & Technology*, pp. 914–917, IEEE, 2009. 23

[15] S. SHIMOJO, G. H. SILVERMAN, and K. NAKAYAMA, "Occlusion and the solution to the aperture problem for motion," 23

[16] C. G. Harris, M. Stephens, *et al.*, "A combined corner and edge detector.," in *Alvey vision conference*, vol. 15, pp. 10–5244, Citeseer, 1988. 24

[17] T. Brox, A. Bruhn, N. Papenberg, and J. Weickert, "High accuracy optical flow estimation based on a theory for warping," in *European conference on computer vision*, pp. 25–36, Springer, 2004. 24

[18] E. Meinhardt-Llopis, J. S. Pérez, and D. Kondermann, "Horn-schunck optical flow with a multi-scale strategy," *Image Processing on line*, vol. 2013, pp. 151–172, 2013. 25

[19] C. W. Gear, B. Leimkuhler, and G. K. Gupta, "Automatic integration of euler-lagrange equations with constraints," *Journal of Computational and Applied Mathematics*, vol. 12, pp. 77–90, 1985. 25

[20] D. Pathak, R. Girshick, P. Dollár, T. Darrell, and B. Hariharan, "Learning features by watching objects move," in *Computer Vision and Pattern Recognition (CVPR)*, 2017. 26

[21] T. Ahonen, A. Hadid, and M. Pietikainen, "Face description with local binary patterns: Application to face recognition," *IEEE Transactions on Pattern Analysis & Machine Intelligence*, no. 12, pp. 2037–2041, 2006. 29

[22] E. Ilg, N. Mayer, T. Saikia, M. Keuper, A. Dosovitskiy, and T. Brox, "Flownet 2.0: Evolution of optical flow estimation with deep networks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 2462–2470, 2017. 45, 60

[23] C. Boettiger, "An introduction to docker for reproducible research," *ACM SIGOPS Operating Systems Review*, vol. 49, no. 1, pp. 71–79, 2015. 54

[24] D. Sun, X. Yang, M.-Y. Liu, and J. Kautz, "Models matter, so does training: an empirical study of cnns for optical flow estimation," *arXiv preprint arXiv:1809.05571*, 2018. 60

[25] P. Liu, M. Lyu, I. King, and J. Xu, "Selflow: Self-supervised learning of optical flow," *arXiv preprint arXiv:1904.09117*, 2019. 60